DOCUMENT RESUME

ED 042 363 24 EM 008 418

AUTHOR	Stankard, Martin F., Jr.
TITLE	Optimal Use of a Computer Based Instruction System in an Existing Urban School District. Final Report. (Optimal Student Scheduling for Computer Assisted Instruction).
INSTITUTION	Pennsylvania Univ., Philadelphia.
SPONS AGENCY	Office of Education (DHEW), Washington, D.C. Bureau of Research.
BUREAU NO	BR-8-B-087
PUB DATE	Apr 69
GRANT	OEG-08-080087-3731(010)
NOTE	247p.
EDRS PRICE	EDRS Price MF-$1.00 HC-$12.45
DESCRIPTORS	*Algorithms, *Computer Assisted Instruction, Models, Predictive Measurement, Programing, *Scheduling, Time Factors (Learning)

ABSTRACT

Criteria were developed for deciding the fraction of course attendance time that students should spend on a computer-assisted instruction (CAI) course in order to maximize the average final achievement of the class, subject to constraints on the probability of individual student failure and on the available console capacity. The major elements of this scheduling procedure are: (1) a model of student learning that relates student time allocations to expected achievement, (2) an objective function and optimization procedure, and (3) a procedure for forecasting each student's learning characteristics. The most statistically reliable learning model was the familiar learning curve or logistic function. A objective function for the scheduling procedure was formulated by systematically reviewing alternative ways of mathematically combining expected test results. The problem of assigning console time was shown to be a non-linear programing problem. Several efforts to find off-line predictors of on-line student learning were unsuccessful; but an on-line selection aid was designed and evaluated. A case is used to illustrate the scheduling algorithm. A bibliography and a glossary are appended. (Author/JY)

ED0 42363

U.S. DEPARTMENT OF HEALTH, EDUCATION & WELFARE
OFFICE OF EDUCATION

THIS DOCUMENT HAS BEEN REPRODUCED EXACTLY AS RECEIVED FROM THE PERSON OR ORGANIZATION ORIGINATING IT. POINTS OF VIEW OR OPINIONS STATED DO NOT NECESSARILY REPRESENT OFFICIAL OFFICE OF EDUCATION POSITION OR POLICY.

FINAL REPORT
Project No. 8-B 087
Grant No. OEG-08-080087-3731(010)

OPTIMAL USE OF A COMPUTER BASED INSTRUCTION SYSTEM IN AN EXISTING URBAN SCHOOL DISTRICT

April 1969

U. S. DEPARTMENT OF
HEALTH, EDUCATION, AND WELFARE

Office of Education
Bureau of Research

EM008 418

This document was processed for the ERIC Document Reproduction Service by the ERIC Clearinghouse at Stanford. We are aware that some pages probably will not be readable in microfiche or in a hardcopy enlargement. However, this is the best available copy, and we feel that the document should not be withheld from interested readers on the basis of these unreadable pages alone.

ED0 42363

Final Report

Project No. 8-B-087
Grant No. OEG-08-080087-3731(010)

OPTIMAL USE OF A COMPUTER BASED INSTRUCTION SYSTEM IN AN EXISTING URBAN SCHOOL DISTRICT

(OPTIMAL STUDENT SCHEDULING FOR COMPUTER ASSISTED INSTRUCTION)

Martin F. Stankard, Jr.

University of Pennsylvania

Philadelphia, Pennsylvania

April 16, 1969

BEST COPY AVAILABLE

The research reported herein was performed pursuant to a Grant No. OEG-08-080087-3731(010) with the Office of Education U. S. Department of Health, Education, and Welfare. Contractors undertaking such projects under Government sponsorship are encouraged to express freely their professional judgment in the conduct of the project. Points of view or opinions stated do not, therefore, necessarily represent official Office of Education position or policy.

U. S. DEPARTMENT OF
HEALTH, EDUCATION, AND WELFARE

Office of Education
Bureau of Research

TABLE OF CONTENTS

Page

LIST OF FIGURES vi
LIST OF TABLES viii
PREFACE ix
ACKNOWLEDGEMENTS xi
SUMMARY xiv

Chapter

One A SURVEY OF THIS RESEARCH 1

Introduction

- The General Problem of Urban Education
- The Introduction of Computers into the Education Process
- Major Problem Areas in CAI Management

Purpose of the Study
Limitations on the Scope of the Thesis
Methodology

- The Use of Models
- Objectives
- Hypothesis Testing

Summary of the Scheduling Procedure
Implications of this Research

- The Conduct of Further Research: Implications
- Implications of this Research for Project GROW Management
- Implications for Management of Existing School Districts

TABLE OF CONTENTS--Continued

Chapter Page

Two A HISTORY OF COMPUTERIZED INSTRUCTION . . 24
  Early Research
  Research by Learning Theorists
  Research on the Management of CAI
  Summary
  References

Three COMPUTERIZED INSTRUCTION: SYSTEM DESCRIPTION AND DEFINITIONS 36
  Introduction
  Elements of the CAI System
  - Curriculum Writing
  - Students and the Learning Process
  - Teachers

  Organization of the Computer System
  - Structure
  - Communication
  - Control

  Assumptions
  Summary
  Appendix 3. 1 - System Data Sources
  Data Elements Used
  References

Four A MODEL OF STUDENT-CURRICULUM INTERACTION 63
  Introduction
  Basic Elements of the Learning Model
  - Index of Mastery
  - Factors Affecting Mastery of Subject Matter

  Student Model I

TABLE OF CONTENTS--Continued

Chapter Page

- Growth of Amount of Material Learned
- A Unit of Instruction
- The Learning Model

Student Model II
Estimating Parameters for Learning Model II

- Statistical Analysis--Model II
- Discussion of Results
- Conclusion

Summary
Appendix 4. 1 - Data Definitions
References

Five THE SCHEDULING PROBLEM FORMULATION 91

Object Function
Allocation Procedure: Development

- The Allocation Problem
- Illustration of Optimal Solution

Properties of the Solution to the Allocation Problem
Appendix 5. 1 - Kuhn-Tucker Condition for the Scheduling Problem
References

Six ESTIMATION OF STUDENT LEARNING PARAMETERS 121

Introduction
An Idealized Data Gathering Procedure

- Types of Variables
- Identifying Variables

An Experimental Data Gathering Procedure
Off-Line Prediction of Learning Model Parameters: First Attempt

TABLE OF CONTENTS--Continued

Chapter Page

Off-Line Prediction of Model II Parameters: Second Attempt
Model II Parameter Estimation Procedure

- Other Properties of Parameter Estimates
  - $\hat{k}_1'$ and $\hat{k}_2$: Bias
- Test of Parameter Estimation Procedure

Summary
References

Seven THE SCHEDULING SYSTEM 157

Introduction
Overview of the Scheduling Procedure
Separable Programming: Implementing the Scheduling Algorithm

- Description of Separable Programming
- Data Preparation Program
- MPS/360 Data Matrix

Infeasible Solution
Final Class Assignment Procedure
Appendix 7.1 - Data Preparation Program
7.2 - Mathematical Programming System Control Program
7.3 - Solution of a Scheduling Problem: A Case
References

BIBLIOGRAPHY 216

GLOSSARY 227

LIST OF FIGURES

A. TEXT

Figure Page

1.1 Overall Schematic of Computerized Instruction System 5

1.2 Schematic of Scheduling Procedure 16

3.1 Overall Schematic of CAI System 38

3.2 Progress on Subject Matter vs. Time on Course . . 46

4.1 Pattern of Instruction 66

4.2 General Form of Learning Curve 76

4.3 Scatter Diagram of Coefficients for Model II 82

5.1 Distribution of Actual Score Around Mean 103

5.2 The Effect of Chance of Failure Constraints on Feasible Schedules 109

5.3 The Effect of Additional Console Capacity on Feasible Schedule 111

6.1 Schematic of the On-line Selection Aid 142

7.1 Schematic of Student Scheduling Procedure 158

7.2 Piecewise Linear Approximation to a Non-Linear Curve 162

7.3 Coefficients for Sample Program Illustration of Linear Approximization Routine for Two Hypothetical Users 165

LIST OF FIGURES--Continued

Figure Page

7.4 Matrix of Coefficients for Sample Problem Mathematical Programming System /360 Separable Programming Data Matrix 167

7.5 Sample Class Schedule 174

7.6 Assignment Tableau for One Week 175

7.7 Example Tableau 2 176

7.8 Example Tableau 3 178

B. APPENDIX

7.3.1 Reduction in Average Grade with Decreasing Console Capacity 208

LIST OF TABLES

Table		Page
4.1	Point Estimates of k_1' and k_2 with Significance Tests	80
4.2	Point Estimates of k_1' and k_2 with Standard Errors	81
5.1	Optimal Schedule: Sample	92
6.1	Sample Correlation Matrix	129
6.2	Estimated Correlation Matrix	137

PREFACE

The first chapter of this thesis is a guide to the rest of it. Chapter One, written in non-technical terms, surveys the problem and the research effort. Chapters Two and Three discuss the development of computer assisted instruction and define the system that has been studied in this effort, respectively. Readers who are interested mainly in technical aspects of this work may omit Chapter Two and much of Chapter Three.

Beginning with Chapter Four, on modeling of student curriculum interaction, the problem of student scheduling for computer assisted instruction is formulated. In Chapter Five the objective and constraints of the problem are defined and a mathematical structure is developed. The problem of estimating parameters required by the mathematical formulation of the problem is discussed in Chapter Six. Finally, Chapter Seven presents a method for solving the mathematically formulated student scheduling problem.

Chapter One--the survey--contains a statement of implications that have been drawn from this effort. In order to make the thesis easier to read, a glossary of most technical terms is included.

References are numbered according to the list at the end of each chapter. Within a chapter, all equations are numbered in order. When equations of another chapter are referred to, the chapter number prefixes the equation number. Thus equation 6. 1 is equation number 1 of Chapter Six.

ACKNOWLEDGEMENTS

My interest in educational management problems was originally stimulated by Professor Roger Sisson, over three years ago. My special thanks go to him for much more than striking the spark, however. Besides supervising this thesis, Professor Sisson has provided continuous guidance and stimulation both as a teacher and as supervisor of my project work on educational management and planning problems.

I also owe special thanks to Dr. Sylvia Charp of the Instructional Systems Department of the School District of Philadelphia. As the manager of Project GROW and as an educator concerned with educational applications of computers her interest and cooperation in this research has been most generous and always helpful.

It is impossible to acknowledge specifically all of the influences that individuals have had upon this thesis; but I acknowledge my great general indebtedness to the statistics and operations research faculty of the Wharton School and to my fellow students there. Important direct and indirect contributions have been made to this thesis by Professors Russell Ackoff, Richard Clelland, John deCani, Shiv Gupta and Donald Morrison.

Many helpful comments and constructive criticisms were received from my fellow students. I particularly acknowledge my gratitude to Charles Goldman, Thomas Jefferson, Christoph Maier-Rothe, Bani Sinha and K. S. Krishnan.

Many Project GROW personnel of the Instructional Systems Department and personnel of the Division of Research--Philadelphia School District, have cooperated in this research. From the Division of Research, Dr. John Hayman, Dr. William Theimer and Mr. James Diamond have all been most helpful. Project GROW personnel: Alex Alexander, Arthur Sandstrom, Selig Schrager and John Woolson have each contributed to this effort. I also thank the biology and reading teachers and administrators at Germantown High School and at Wanamaker Junior High School.

Mr. Roger Wye and Mr. Stanley McConnell of Philco-Ford have also been most helpful. Mr. McConnell was especially patient in answering my questions about the Project GROW computer system and systems programs.

This thesis has been supported by a grant from the U. S. Office of Education--Small Projects Office of the Bureau of Research and by a grant from the Philco-Ford Corporation. I am greatly indebted to both of these organizations for their financial help.

The staff of the Management Science Center--Wharton School has been more than cooperative in caring for the administrative details of this research effort. I am greatly indebted to Mrs. Elva Power who worked diligently to convert the various drafts of this thesis into a finished document.

I personally assume full responsibility for any errors that may remain in this thesis.

Martin F. Stankard, Jr.

SUMMARY

Student selection and scheduling for computer assisted instruction is a hitherto neglected problem in managing the operations of a CAI system. This thesis develops a basis for deciding the fraction of course attendance time that students should spend on a CAI course in order to maximize the average final achievement of the class subject to constraints on the probability of individual student failure and on the available console capacity. The major elements of this scheduling procedure are: (1) a model of student learning that relates student time allocations to expected achievement, (2) an objective function and optimization procedure, and (3) a procedure for forecasting each student's learning characteristics.

Two learning models are presented, one uses finite difference equations to relate successive periods of instruction and testing, the second is the familiar learning curve or logistic function. Both were tested with learning data gathered by the Philadelphia School District Project GROW computer system. These data were from high school students on a computerized biology curriculum. The finite difference equation model was rejected and the logistic model was found to be a statistically good description of the growth of learned material as a function of cumulative console instruction time.

An objective function for the scheduling procedure was formulated by systematically reviewing alternative ways of mathematically combining expected test results. The final objective was to maximize the average forecasted class achievement on a final test, subject to constraints on the probability that any student might fail. Other formulations included: unconstrained maximization of class average achievement and unconstrained maximization of the probability that all students should score above a passing grade.

The problem of specifying console time assignment for each student in the group under consideration is shown to be a non linear programming problem. Separable programming by means of the Mathematical Programming System/360 package is used to find the console time schedule that satisfies all constraints and that optimizes the expected class average. The problem of infeasible schedules is discussed in terms of the information provided by the mathematical programming algorithm.

Several approaches to the forecasting of student learning characteristics are reported. Two regression studies are presented.

An experiment which used a sample of programmed instruction to forecast subsequent student learning at consoles is also discussed. All of these efforts to find off-line predictors of on-line students were unsuccessful. An on-line selection aid for gathering the required information on student learning characteristics is designed, and some of its statistical properties are examined.

A case is used to illustrate the scheduling algorithm.

CHAPTER ONE

A SURVEY OF THIS RESEARCH

INTRODUCTION

The General Problem of Urban Education

All is not well with urban primary and secondary public education. The schools are supposed to impart basic skills and social values to young entrants to society; yet many symptoms of failure or disorder are apparent. Dropout rates in most urban areas are high, urban school systems are generally in a state of financial crisis, and most objective measures of students' performance are far below desired levels.

The general problem is an economic one. With the present techniques of teaching, there is a widening gap between demands made on urban educational systems and the economic resources available to city school systems. Society seems to demand that all children be able to read and write up to some standard. Yet the present system may not be able to meet even this basic demand with the resources it can acquire. Or in trying to meet the demands for "equality of educational opportunity" society may decide that just as a college education is not now the right of all, so too, real literacy for all may be an unattainable goal.

A second aspect of the problem of urban education is, broadly stated, the question of quantity versus quality. The most casual observer is aware of the dramatic increases in quantity or quality of production in most goods and services demanded by our society. Similar dramatic increases in per employee productivity (either quantitative or qualitative) have been lacking in educational systems. Education has been in the past, and still is, one of the most labor intensive activities of our economy.

There is some hope for solving the general problem, however. Economic history shows that as demand for a labor intensive service increases, innovators are stimulated to seek out a more capital intensive way of satisfying the demand. The effect of this has usually been a substitution of capital for labor, but this is not sufficient for closing the gap. Together with increased capital inputs to the education process there must be a parallel increase in the efficiency with which both labor (teachers) and capital (machines) are employed. This means that the right combination of labor and capital equipment together with procedures for combining them into a more productive whole must be found.

Where the innovation process has been at work the results are truly astonishing. For example, one man with a tractor can plow as much ground in a day as a large number of men can plow by hand in the same time. In like manner innovators are investigating the use of specially programmed digital computers to carry out some functions of education. The development of effective management procedures, however, is not widely emphasized by these innovators. And far more attention must be paid to the management of capital facilities in schools whether the equipment happens to be film or slide projectors, tape recorders, elaborate video tape and television equipment or now, computers. One motive for this thesis is the belief that without careful attention to problems of management and control, virtually any innovation can be rendered valueless or worse. This thesis concentrates on one such management or control problem that is important in adapting computers to instructional uses. We are concerned with how to combine student time and the services of a computer assisted instruction system (CAI system) to achieve the desired output most effectively.

The Introduction of Computers into the Education Process

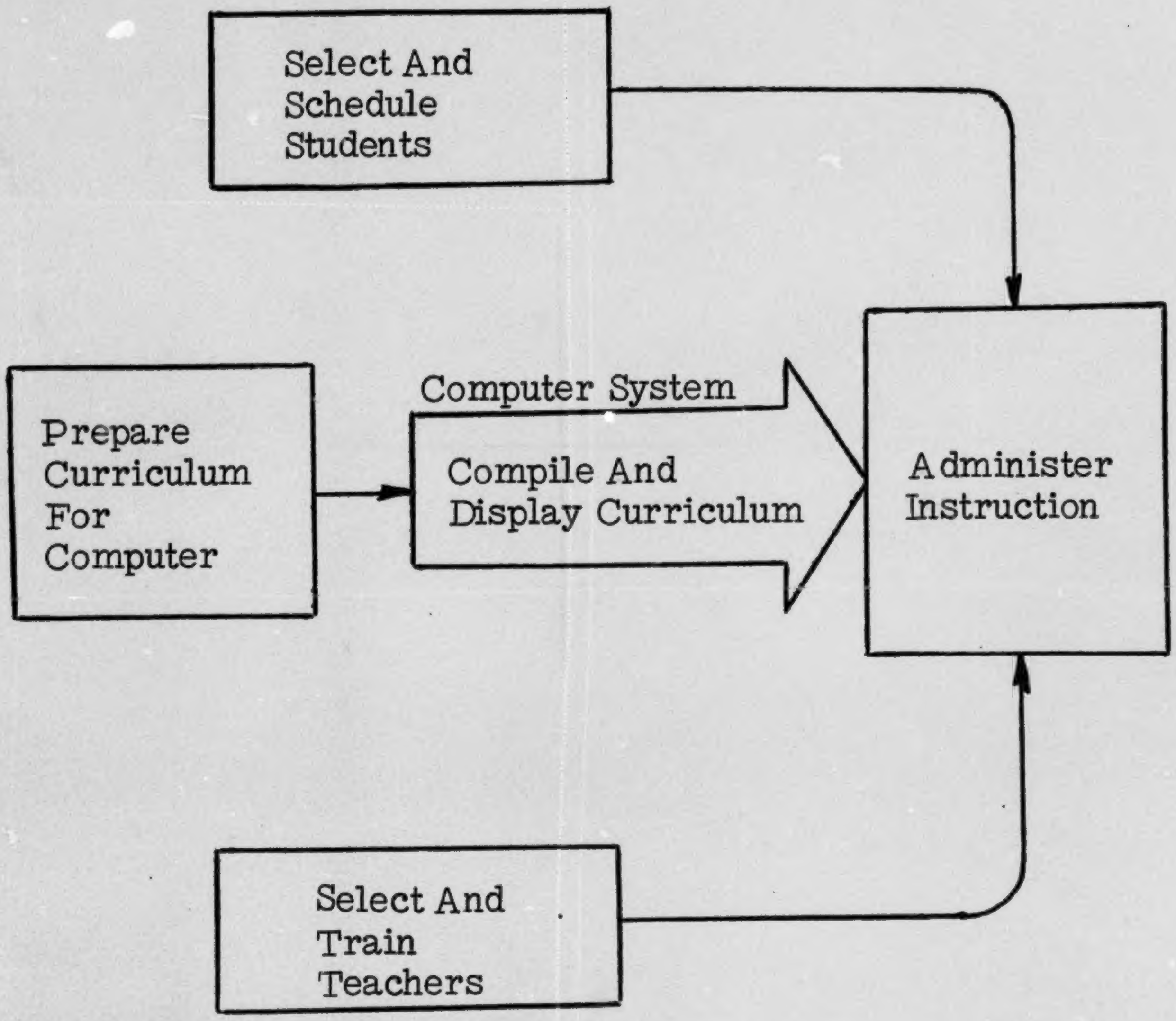
Exactly what functions of education can be or should be computerized is a moot question, and it is only discussed briefly here. Some commentators hold that to substitute a machine for a human teacher is to cheat the student. Others appear to hold to a relationship of the type "computer = calculator = numbers = cannot teach ... history or art or humanities." That is they tend to erroneously equate related but hardly synonymous elements. The future extent of computerization of the teaching process cannot be predicted now with much reliability.

The Philadelphia School District is operating an experimental computer assisted instruction (CAI) system referred to as Project GROW. The name GROW stands for the four schools that are participating, Germantown, Roosevelt, Overbrook and Wanamaker. Like most CAI systems Project GROW uses digital computers to guide student performance on a specially prepared curriculum. Based on student responses, the computer has been programmed to display different elements of curriculum to the student.

Although exact operating cost figures are not yet available, the system is several times as expensive (per student per hour) to operate as it is to teach a student by traditional means. The expense of using the system highlights the need to manage it effectively, so that it may be evaluated both for its educational productivity and for its economic consequences.

Major Problem Areas in CAI Management

Figure 1.1 shows the three main elements in a CAI system; (1) students, (2) the computer system equipped and programmed to instruct students and (3) teachers. Management of an operating CAI system must make decisions that affect each of the three. The student scheduling and teacher training problem areas have been given the least attention by researchers.


Select And Schedule Students
Prepare Curriculum For Computer
Computer System
Compile And Display Curriculum
Administer Instruction
Select And Train Teachers

Overall Schematic of Computerized Instruction System

Figure 1. 1

This study is centered on the topmost block of the overall schematic, the selection and scheduling of students for computerized instruction. Briefly, the problem is to determine how long each of a group of students should be assigned to computerized instruction. The scheduling problem includes the selection of students for computerized instruction. Any student scheduled for a negligibly small amount of computer time has in fact been selected for non-computerized instruction.

PURPOSES OF THE STUDY

The purposes of this study are twofold. The first is to formulate and analyze a specific problem in the management of an extremely complex educational system--because the problem itself is a critical one. Second, the thesis aims to illustrate, by example, the usefulness of a scientific operations research approach to complex management problems in education.

The results of this thesis should provide a framework which managers of CAI facilities can use to consider student scheduling decisions. The framework provides guidelines for making these decisions; not necessarily commandments. The number of factors which are variable in each specific case as well as the lack of a tested quantitative theory of the student learning process means that detailed and widely applicable results are not to be expected at this time.

Demonstrating the usefulness of the operations research approach to educators may ultimately be the more important purpose. By seeing examples, managers in educational systems may call on operations researchers more often to help formulate and solve crucial problems in urban education. I firmly believe this to be good both for operations research as a growing profession and for educational managers faced with increasingly complex organizational problems.

LIMITATIONS ON THE SCOPE OF THE THESIS

The scope or generality of this thesis is dictated by its purposes, which we have just discussed, and by limitations on data and other resources. The operating problem studied is narrow in scope. But, the methods used in studying this specific operating question are very general, and with sufficient resources these methods apply to other critical problems of educational management.

Our concern is the management of students during the computerized portion of a course. Although the course may have non-computerized elements, the criterion that will be used is performance on the computerized (on-line) instruction. We assume that the part of a course which is not computerized can be taught by the teachers in the student time remaining after console time is scheduled. This restriction in scope is made for several reasons. First, in scheduling students for CAI we may always put an upper limit on the time that anyone spends on a console and thus insure a minimum of time with a regular class. Second, preliminary research findings indicate that available CAI instruction in many subjects is superior (in terms of student achievement) to the off-line alternative. Third, a properly trained teacher is sufficiently flexible to tailor the non-computerized segments of the course to the needs of the students. Finally, data which will be discussed in Chapter Six indicate that for students who receive computerized instruction, but have different teachers, there is no significant difference in performance on an achievement test administered off-line (i.e., not by the computer).

Although the methods and some of the models of this thesis are developed from general assumptions, the Project GROW system has served as the data base for detailed research. This narrows the scope of the empirical portions of the study. Within Project GROW two courses, reading and biology, were being presented while this research progressed, and data gathering has been restricted to these two subjects. Of the four schools, the analyses of data have been limited to observations made at Wanamaker Junior High School and Germantown Senior High School. Most detailed analyses have been made on data from the Germantown Biology course for spring and fall 1968.

The scope of the analytic portions of this thesis is restricted by the assumptions on which they are based. From these assumptions the optimality of a schedule of student instruction can be deduced logically. If these assumptions were changed, however, the results presented here would probably not be applicable. For example, the measure of an individual student's performance is to be his score on a test covering the computerized portion of the course. It is natural to suggest that the performance measure should include the results of the non-computerized portion of the course. This extension, however, would add a new dimension of tremendous complexity to the problem. The problem is already formidable in its restricted form.

METHODOLOGY

The Use of Models

An essential characteristic of operations research and of science in general is the use of models of the problem situation of interest. In this study mathematical models are used as proxies for parts of the real system being studied. When alternative models may be used to represent a phenomenon, the principle or parsimony helps in selecting one. There is a distinction between the building of a model and the testing of hypotheses. Building a model may involve deriving and testing hypotheses, while testing hypotheses does not imply building a model.

The modeling approach organizes our assumptions, hypotheses and facts in an orderly way so that their dependencies can be seen clearly. Once these bits and pieces are assembled we can often derive testable hypotheses from the models. Wherever possible it is best to use tested hypotheses (or facts) in building models to be used in solving a problem. Where this is not possible, the hypotheses suggested by the models should be tested against the actual system which the model represents.

There are a few specific requirements which a model of a problem situation must satisfy. Such a model should relate the value of the output of a system (the objective or measure of performance) to some policy variable (controllable variable) and other relevant factors (uncontrollables). The formulation of a model should also include significant constraints on allowable policies--for example in scheduling students, limits on the amount of computer time available must be expressed as a constraint.

The models used in this thesis rest upon several assumptions. As far as practicable in the research the most important assumptions have been tested. This poses a practical problem. Occasionally, an assumption that appears reasonable may not turn out (upon testing) to be in complete agreement with some of the available evidence. When this happens the options are to test the assumptions further or to scrap them and develop a new model on different assumptions. The practical problem is that research time and data are limited, and the value of further developing and testing a particular hypothesis must be weighed against alternative uses of the available time and effort.

Objectives

One of the more controversial problems in operations research is the choice of the objective. This problem is complicated since in the educational area there may be several conflicting sub-objectives considered in evaluating the instructional output of a system. The selection of a final objective or measure of performance must be done heuristically. A preferred measure of system performance is defined by scrutinizing several alternatives. In arriving at a performance measure for use in this research, five alternatives have been examined in detail. The objective, developed specifically for this problem, converts some of the sub-objectives into constraints on allowable decisions.

The search for an acceptable objective used three criteria of acceptability in order to judge alternative performance measures. These were: (1) The performance index should not obviously cause any group of students to be excluded from computerized instruction,

(2) the performance measure should be easily interpretable by the educators responsible for the instructional system, and (3) the performance measure shall be operationally definable and translatable into an equivalent mathematical function so that mathematical optimization theory may be used to improve performance.

Without going into much detail here the five are:

(1) Maximize the average achievement score of a class of students.

(2) Maximize the average achievement score of the class minus some function of the variance of the class scores.

(3) Maximize the α percentile of the class grade distribution, where $\alpha < 0.50$ (Fractile Criterion).

(4) Maximize the probability that all n students score above some passing grade, i.e., that all students pass.

(5) Maximize the average class achievement score subject to a constraint on the probability that each student might fail (fall below a critical score).

These objective functions are discussed in detail in Chapter Five.

The fifth alternative, above, is the mathematical formulation of the objectives of the educator. It can be converted to a mathematically solvable model of the scheduling problem.

Hypothesis Testing

From the modeling phase of the research process many explicit hypotheses are developed. Some of these hypotheses deal with sub-models of portions of the overall problem; others are more traditional tests on the value of some parameter in a model. From a theoretical and an empirical point of view the most important sub-hypotheses are concerned with the relationships between a student's past performance (indicated by tests and attendance records) and the pace and effectiveness of his use of computer instructional time. The techniques used to test these sub-hypotheses vary depending upon the form of each hypothesis.

In testing models of student learning we propose a particular theoretical functional form and test its goodness of fit to data. If the function is in agreement with the data, we may have reserved confidence in the underlying theory. Now consider the second set of sub-hypotheses. No tried theory exists for predicting the characteristics of a student's CAI learning behavior on the basis of his historical data. So hypotheses about relationships between past performance and future CAI interactions are stated in terms of statistical dependencies among variables or as linear statistical models in some cases. The tests of these hypotheses or models do not necessarily provide any theory which explains the dependencies which exist. Thus the long run solution to this type of student performance forecasting problem is the development of an explanatory theory of human learning. This theory is the goal of all learning theorists, and so far success is no where in sight.

In summary, the methodology applied in this thesis has been similar to that used to solve managerial problems in the industrial and military sectors. The use of an operations research methodology to formulate and analyze a complex problem in educational management is a relatively new application. The methodology uses models in place of the real system. The objectives of the decision-maker are defined, the constraints on his actions are identified and these are organized into a structure which makes it possible to consider alternative management policies without greatly disrupting the real system.

SUMMARY OF THE SCHEDULING PROCEDURE

One result of this research is a scheduling procedure that is illustrated schematically in Figure 1.2. It is difficult to discuss the scheduling problem and its solution procedure in isolation since there is much behind the development of the final formulation.

The details will not be discussed in this summary; we will instead briefly present the final problem formulation and the scheduling procedure in order to suggest the direction of the research.

The problem formulation phase of this research has led to the problem below:

Objective

Maximize (with respect to individual student time schedules) the average final test score for all students being considered.

Constraints

Subject to the constraints:

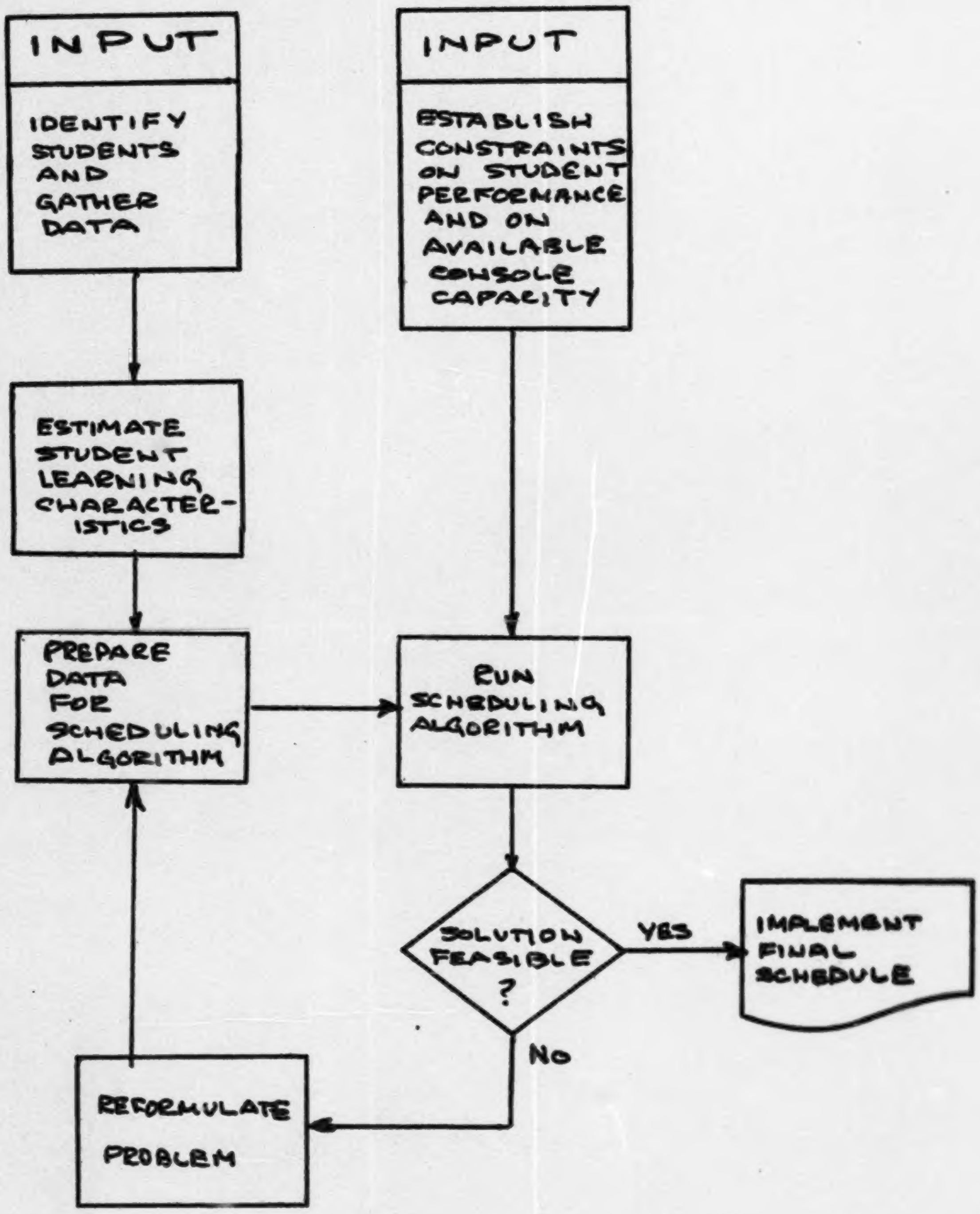
Each student shall have a chance no greater than α ($0 < \alpha < 1$) of scoring below a designated passing grade on the final test.

and

There shall be enough console time to implement the final schedule.

There is an activity labeled, "run scheduling algorithm," in Figure 1.2. When the data has been properly prepared the scheduling algorithm either solves the above problem or it reports that there is no solution that satisfies constraints.

The steps leading up to the use of the scheduling algorithm are mainly data processing. The most important start with the INPUT block labeled "identify students and gather data." For each student being considered for computerized instruction the scheduling algorithm requires an estimate of that student's learning characteristics.


INPUT
IDENTIFY STUDENTS AND GATHER DATA
INPUT
ESTABLISH CONSTRAINTS ON STUDENT PERFORMANCE AND ON AVAILABLE CONSOLE CAPACITY
ESTIMATE STUDENT LEARNING CHARACTER-ISTICS
PREPARE DATA FOR SCHEDULING ALGORITHM
RUN SCHEDULING ALGORITHM
SOLUTION FEASIBLE ?
YES
IMPLEMENT FINAL SCHEDULE
NO
REFORMULATE PROBLEM

Schematic of Scheduling Procedure

Figure 1. 2

In concrete terms, we must estimate k_1' and k_2 in a learning model of the form

$$\text{expected test score after time } (t) = [1 + e^{-k_1't - k_2}]^{-1} \quad . \qquad (1)$$

Generally each student will have a different pair of k_1', k_2. Several methods for estimating the parameters have been tried with varying success; for example, the use of regression techniques to predict k_1' and k_2 from each student's school records. Typical prediction variables would be student reading or verbal ability, arithmetic and grammar achievement and attendance data.

Once values for k_1' and k_2 are estimated for each student from data on that student, the relationship between total course time on console (t) and expected final test score is defined. Since these relationships are non linear (equation 1) they are approximated by a series of linear functions. In this way the original non linear scheduling problem can be solved by the procedure known as separable programming. The simplex algorithm (with certain modifications) is used to solve the linearized problem of maximizing the class average subject to capacity and failure probability constraints.

After using the scheduling algorithm to find an optimal feasible solution to the mathematical problem, console assignments can be made by another simple algorithm. If a feasible solution does not exist the problem must be reformulated in one of several possible ways.

IMPLICATIONS OF THIS RESEARCH

The Conduct of Further Research: Implications

One earmark of good scientific research is that it should help provide for the improvement of future research on related problems. It is impossible to express in detail all of the lessons learned in carrying out this study. The most evident lessons, of course, are those where some type of error was made or where some plan or design failed.

In carrying out research in the urban school district environment a management scientist is utterly dependent upon dozens of people (and where students are involved, even hundreds). When these people do not perform as expected, the conduct of the research may be affected. The experience in this study has been that the people who must be relied upon almost never really perform exactly as they are asked to or even as they agreed. In the sensitive data gathering phase of the research every effort should be made to spot check data collection and to provide redundancy in identifying data elements. In gathering the data from the computer system seemingly trivial oversights by technicians can cause major losses of data.

Many problems encountered in this research arose simply from the newness of the system being studied and the fact that there had been no previous research work performed on it. Other problems, however, are apparently a fact of life in the school district environment. An example is the problem of students who transfer in and out of courses before any significant data can be gathered on their learning behavior. There is a tremendous decrease in sample sizes because of these class changes and other similar factors. The general rule is to take as large a sample as can possibly be managed, then analyze the data immediately so that any errors are found while there may still be a chance to go back and do a follow-up study on what went wrong.

The usefulness of the computer in gathering data on student learning is hard to over-estimate. Our ability to use the computer to gather the right data on the right students is not nearly as well developed.

Implications of this Research for Project GROW Management

Two classes of problems confront Project GROW management (a) startup problems arising from the newness of the computer based teaching system and (b) operating problems that may be solved routinely by specially developed procedures. A prime implication of this research is that both of these classes of problems (but particularly the operating questions) will require very substantial resources for solution. Thorough and purposeful management of CAI will put much greater burdens on administrative manpower than do the inherently flexible traditional teaching systems.

For example, a teacher cannot know the effect on each student of each minute of classroom instruction. For a teacher this information would be more of a burden than a help. Yet with CAI not only is this type of data gathered for each instructional item and for every student that interacted with that item but it is in a form ideally suited for inexpensive processing by electronic computer. This information is a vast resource. If it is utilized it can help curriculum designers identify portions of a course that are least effective. But, if ten percent of the items in a curriculum of 10,000 items are defective it is a big job to modify them. Although the teacher cannot possibly evaluate every detail of her classroom behavior, with CAI this is possible.

In the research area of student scheduling the same type of logic applies. At present, students are randomly selected from those who are eligible for a particular CAI course. If this is the standard procedure for selection, the rationale underlying it might be that: CAI is good for students, and since the existing system cannot accommodate all eligible students, each should be given an equal chance of having computerized instruction. After reflecting on that proposition we might conclude that it is virtually certain that CAI is not an equally good way to teach all students. We surely would expect some students to have a comparative advantage over others, even on a well designed CAI curriculum. While random selection is a possible solution to the student selection problem, its rationality may be challenged when it is compared with alternatives.

There is a further implication of this research for Project GROW management. The various aspects of an operating CAI system are all so tightly coupled that improvements in one area, such as selection of students, may interact with other areas. For example, if parts of a curriculum are not suited for some identifiable group of students we may either modify the curriculum or we may use this information in the selection process. Changes in each part of the system must be considered for their effects on other parts.

Once a computer curriculum is prepared, its flexibility, or ability to cope with various student behaviors, is fixed. The program can be adaptive to the needs of the student users if that is the way it was originally designed, but the student selection procedure can control the mismatch between students and curriculum and keep it within desirable limits. When decisions on selection and scheduling are made, they should be made with the knowledge of how each student is likely to benefit and in the light of overall system objectives and constraints.

Implications for Management of Existing School Districts

Experience with the rather narrow problem of student scheduling on CAI does provide an opportunity to draw broader implications for management. The main implication is about the attitude that educational management has toward experimentation with computerized instruction. Computer assisted instruction brings together learning theory, psychology, computer-time sharing technology into an extremely complex assembly of men and computers. The attitude of management toward this new complex should be to define the proper role of these systems. This will require a more sophisticated research activity than has been used in the past; in order to allow researchers to influence experimentation with these systems and to enable researchers to develop effective procedures for managing the new systems.

The development of computer hardware and software for teaching may be easier than the development of effective ways to manage the systems. The fact that CAI systems permit students to progress at their own (widely varying) rates will put pressure on management to recognize the varying demands that students make on a teaching system. The data that CAI systems can provide on each student's learning behavior (or lack of it) can be used to refine routine educational decision-making and truly bring learning under the educators' control.

A final implication or educated guess, perhaps, is that the time horizon that management should take in these matters should be a relatively long one--ten to twenty years. They are forced to do this when it comes to building schools. A teacher's salary in the first new years of his career hardly can repay him for the cost of his education. This is also the case with CAI. We might have to be willing to regard these systems as the occasion for educating in the management of the teaching process as well as the means to teach students.

CHAPTER TWO

A HISTORY OF COMPUTERIZED INSTRUCTION

During the 1950's research into the use of Skinner's operant conditioning theory for teaching human subjects mushroomed. Skinner defines an operant as behavior which is emitted by an organism; the operant is often defined as the behavior which leads to a reward. The elemental producers of learning are called the contingencies of reinforcement by Skinner and they are: (1) the occasion on which a behavior occurs, (2) the behavior itself, and (3) the consequences of the behavior [8]. Skinner maintains (and experiments support) that when the three contingencies of reinforcement are properly arranged the behavior of a subject can be modified. For the first time, Skinner had deduced an instructional scheme from his experimental analysis of behavior in animals.

Many programmed text books were developed on Skinnerian principles and several types of teaching machines, based upon programmed instruction and earlier test giving machines, were developed. The armed services' needs for training greatly stimulated this research and development effort. None of these instructional systems, developed before 1958, used a digital computer for monitoring the student and for controlling the instructional presentation. Historically, however, this early research on programmed instruction and mechanical tutoring devices led directly to research on the use of digital computers in the instructional process.

In a few cases analog computers had been used to simulate a particular environment to facilitate training. In the LINK Trainer, a complex system to train aircraft pilots, an analog computer simulates the aircraft. The trainer receives student responses and presents him with stimuli such as would be obtained from flying a real plane. The computer makes no attempt to analyze the correctness of student responses nor to present educational material.

Research on computerized instruction evolved in two phases, and it now appears that research is entering a third phase. At first the research was aimed at demonstrating the technical feasibility of using an electronic computer to control the process of teaching a specific behavior to a human subject. The second phase has been the use of computer based instruction systems by mathematical psychologists and learning theorists to probe the learning behavior of human subjects. The third phase, just beginning, consists of examining the operational and economic feasibility of using these systems to satisfy the actual needs of "real world" educational systems. In this phase the problems of managing a large scale computer assisted instruction system must be studied.

EARLY RESEARCH

Computer assisted instruction (CAI) had its origin in 1958 when a group at the IBM Research Laboratory built a simulator of a Skinner teaching machine. The Electric Typewriter Division of IBM had a contract with the learning psychologist, Skinner, for the development of a teaching machine based on his patents. A simulator of this machine was constructed by Gustave Rath, Nancy Anderson and R. C. Brainerd from the Psychology Department of the IBM Research Laboratory. It taught binary arithmetic [6].

In 1960 an IBM 650 computer was outfitted for courses in stenotyping, German and statistics. These courses were developed at the IBM Research Laboratory by W. Uttal, W. Koppitz, and R. Grubb. In addition to this effort, the University of Illinois, and the Decision Sciences Laboratory at Hanscomb Air Force Base, (with Bolt-Beranck and Newman, Inc.) undertook separate studies of the technical feasibility of computerized teaching systems. By 1961 all of these efforts had demonstrated the feasibility of the general type of computer based instruction system Project GROW now uses [6].

RESEARCH BY LEARNING THEORISTS

The investigation of the theoretical relationships between human learning and computer based instruction then began developing in the 1960's. The main use of computer based instruction in most of this research was to gather detailed data on student learning. This data is used in testing models of learning. Ultimately the validated models could be used in designing optimal teaching systems.

The largest effort in this area has been worked at Stanford University where Richard Atkinson and Patrick Suppes have been studying the teaching of mathematics and reading since 1964. While the study is in part motivated by the need to develop curriculum material for a specific computerized instruction system, it is basically concerned with the development of "optimization models for learning" [2]. This usually means that the problem of presenting instructional messages to maximize the probability of correct responses can be formulated as a multistage decision process and can be solved, for example, by dynamic programming.

If the point ever comes when each piece of information presented to a student has been selected through the solution of a dynamic programming problem of even a few stages we will truly need the capabilities of the electronic computer. At this time, however, it is virtually certain that the vast computing resources which would surely be expended with such an approach are not going to produce commensurate improvements in the student's learning.

In addition to the work at Stanford, considerable research is being done at the University of Illinois and at System Development Corporation (SDC). Karush and Dear at SDC have published findings on optimal strategies for presenting frames to students. So far their results have not been conclusive [1]. That is, basing the presentation sequence of a set of frames on a strategy derived from a detailed model of student learning has not been conclusively shown to be superior to random sequences in the case studied.

The preceding illustrates the relevant types of research being carried out by educational and mathematical psychologists. A more exhaustive survey would almost be a book in its own right and would go beyond the purpose of this thesis. The experience accumulated by researchers, experimenters, and computer manufacturers has now led to application of CAI in the real world of military training and public education. The latest phase of research in CAI has been stimulated by these applications.

RESEARCH ON THE MANAGEMENT OF CAI

At the same time as the work of the learning researchers goes on there is a developing body of research on management decision-making problems posed by the CAI systems. Essentially, the decisions which are made in the context of computerized instruction may be classified as either tactical decisions or strategic decisions.

One of the earliest efforts at the tactical level was a doctoral thesis done in 1962 by Richard Smallwood [9] A Decision Structure for Teaching Machines. He considered the mathematical structure of a mechanical tutoring device which, theoretically, can adjust the instruction to take advantage of learning characteristics of each student, and which can use results of previous instruction to improve its choice of presentation to individual students.

The fact that the decisions which Smallwood was analyzing were to be made during the course of instruction indicates the tactical nature of the work. The approach which is used in this thesis is related to Smallwood's, however, the problem is strategic. This thesis is concerned with the pre-selection and scheduling of students for CAI, based upon information that can be gathered on them before they are exposed to the computer system. The effort here does use the learning characteristics of individual students in making the decisions. This is the main similarity with Smallwood's approach.

There are broad classes of strategic problems which must be solved as the number and extent of CAI applications grow. Although some attention has been given to the problems of the computer system, too little has been aimed at the selection of students and the selection and training of teachers to use CAI. The fact that CAI individualizes the student's learning to reflect his own characteristics also places a greater burden on the administration of an educational system. Students can be in many more states of education than before, and they must be described in more terms than when the teaching process is uniform for all of the students.

Operating CAI systems in Philadelphia (Philco-Ford) and in New York (Radio Corporation of America) as well as others are highlighting management problems which can only be solved by types of analysis that are not usually found in public elementary and secondary education.

This thesis goes beyond existing work for at least two reasons. First the learning characteristics of each individual student are considered when the detailed decisions are made about scheduling of students onto the system. Secondly, the performance criterion which is used in selecting and scheduling students considers the distribution of achievement of all of the students. As the following review of economic and operational studies will show, these two factors have not been studied before.

Kopstein and Seidel [3] at the Human Resources Research Office of George Washington University have been actively studying the economics of computerized instruction relative to traditional instruction. The main conclusions of the study are that without regard to relative effectiveness the estimated cost of CAI at the elementary and secondary level is $3.73 per student console hour vs. a cost of $0.36 per student hour of traditional classroom instruction, and that the growth in cost of traditional instruction (due to teacher pay increases, etc.) together with the tremendous drop in per student curriculum development costs (as the number of students that use the curriculum rises) will rapidly bring the two dollar figures for CAI and traditional instruction to parity.

One factor neglected by Kopstein and Seidel is the fact that one hour of console time is purely instructional and that an hour with a teacher (especially in a present day urban high school) may only be partly devoted to instruction. The teacher attends to many administration and disciplinary matters which do not arise when the students are at consoles. The console can attend to administrative data gathering without reducing the student's instructional exposure.

At U.C.L.A. in the early 1960's Arnold Roe and Harry W. Case carried out studies of the economic consequences of alternative teaching systems at the university level. They were not primarily concerned with computerized instruction. They did regard CAI as an alternative but only if its data gathering and processing capabilities were used together with systems for making the teaching process "adaptive" [7].

The operating CAI system in New York has stimulated work there by Randall and Blaschke. They describe an economic analysis procedure which includes relationships among:

- per student cost of capital equipment (including CAI equipment)
- cost per unit of student achievement increase
- utilization of capital facilities.

So far Randall and Blaschke do not explicitly include student achievement in their formulations [5]. Their stated economic analysis procedure appears to be a description of an ideal, rather than the actual.

SUMMARY

The history of research on CAI has developed from the origins of research on programmed instruction and automated tutoring devices. Research has progressed from first feasibility studies, through the development of working computer assisted instruction systems. The learning psychologists, largely from universities, have made CAI a research tool for exploring learning in human subjects. The attempts by school districts, and other education and training organizations, to use CAI as a practical way of providing instruction, are pointing up the management problems.

The relationship between this thesis and other research in CAI has been discussed. The management problems of concern here are those which arise before a group of students has been chosen for CAI. This research does use information on the students learning characteristics to make decisions about their needs for time on a computer assisted instruction system.

REFERENCES

Chapter Two

[1] Dear, R. E.; Silberman, H. F.; Estavan, D. P.; and Atkinson, R. C. "An Optimal Strategy for the Presentation of Paired-Associate Items." Behavioral Science, Volume 12, No. 1, January 1967.

[2] Groen, G. J., and Atkinson, R. C. "Models for Optimizing the Learning Process." Technical Report No. 92, February 11, 1966, Institute for Mathematical Studies in The Social Sciences, Stanford University, Stanford, California.

[3] Kopstein, Felix F., and Seidel, Robert J. "Computer Administered Instruction Versus Traditionally Administered Instruction: Economics." A.V. Communications Review, Volume 16, No. 2, Summer 1968, pp. 147-175.

[4] Niedermeyer, Fred C. "The Relevance of Frame Sequence in Programmed Instruction: An Addition to the Dialogue." A.V. Communication Review, Volume 16, No. 3, Fall 1968, pp. 301-317.

[5] Randall, R., and Blaschke, C. "Economics, Management and Public Policy." Educational Technology, Volume 8, No. 12, June 30, 1968.

[6] Rath, Gustave J. "The Development of Computer Assisted Instruction." I. E. E. E. Transactions on Human Factors in Electronics, June 1967.

[7] Roe, Arnold. "An Adaptive Decision Structure for Educational Systems." Los Angeles: Report No. 63-63, Department of Engineering, University of California, Los Angeles, 1962.

[8] Skinner, B. F. The Technology of Teaching. New York: Appleton Century Crofts, Division of The Meredith Corporation, 1968.

[9] Smallwood, Richard D. A Decision Structure for Teaching Machines. Cambridge, Massachusetts: The M.I.T. Press, 1962.

[10] Wilson, H. A., and Atkinson, R. C. "Computer Based Instruction in Initial Reading: A Progress Report on the Stanford Project." Technical Report No. 119, August 25, 1967, Institute for Mathematical Studies in The Social Sciences, Stanford University, Stanford, California.

CHAPTER THREE

COMPUTERIZED INSTRUCTION: SYSTEM DESCRIPTION AND DEFINITIONS

INTRODUCTION

Computer assisted instruction (CAI) is a new type of teaching system. Although the means of teaching in CAI are radically different, the aim of the process is much the same as in the traditional teaching system. The aim is to give the student an understanding of some subject matter. After a period of computerized instruction, the student is given a test. A student who can correctly solve the problems and answer the questions on the test is said to have learned the subject matter.

Sitting at a console equipped with a cathode ray tube, a light pen, and a typewriter-like key board, the student interacts with the computerized curriculum without outside intervention or control. His responses are processed immediately. All flows of information about curriculum and about student responses are handled internally by the computer system.

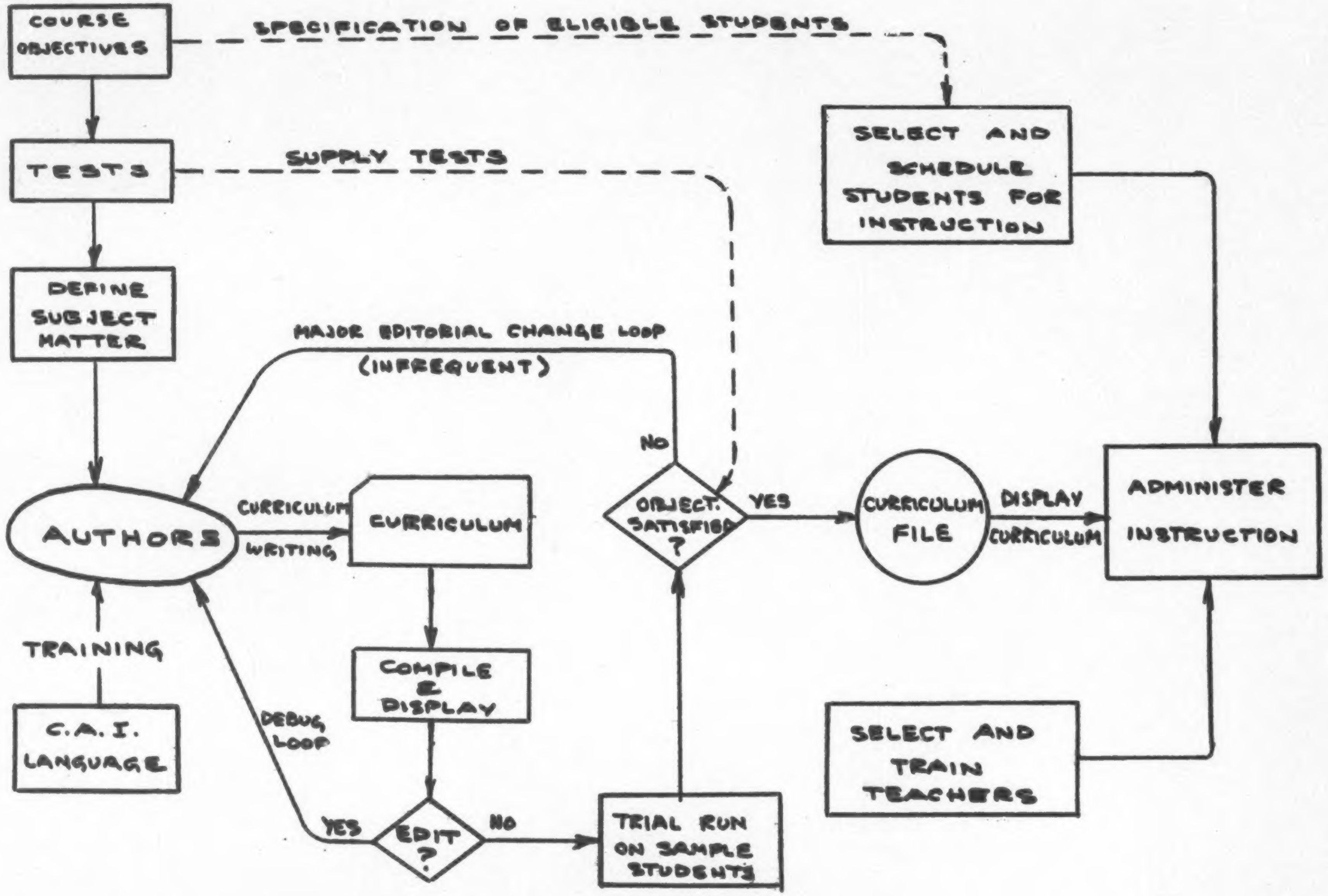
Focusing on Philadelphia's Project GROW, our problem here is to determine which students should be assigned to the CAI consoles, and for how long. In Project GROW a computer system is being used to teach junior and senior high school students biology and remedial reading. There are more students than we can possibly accommodate, so we want to find a rational basis for allocating the use of the instructional system. The remainder of this chapter describes and defines the system and the terms which will be used in selecting and scheduling students. The final section summarizes the assumptions which are used later.

ELEMENTS OF THE CAI SYSTEM

There is more to a CAI system than simply a computer system--other elements are the curriculum and curriculum writing process, the students and the means for selecting them, and the teachers trained to work with CAI. Figure 3. 1 below shows schematically how these three factors; students, teachers, and curriculum come together in the instructional process.

1) Curriculum Writing

Curriculum writing proceeds in this way: the core of the process is the subject matter to be taught. This material is identified in the objectives of the program or course of instruction. From the course objectives, achievement tests are developed for the


COURSE OBJECTIVES
SPECIFICATION OF ELIGIBLE STUDENTS
TESTS
SUPPLY TESTS
DEFINE SUBJECT MATTER
SELECT AND SCHEDULE STUDENTS FOR INSTRUCTION
MAJOR EDITORIAL CHANGE LOOP (INFREQUENT)
NO
AUTHORS
CURRICULUM WRITING
CURRICULUM
OBJECT. SATISFIED ?
YES
CURRICULUM FILE
DISPLAY CURRICULUM
ADMINISTER INSTRUCTION
TRAINING
C.A.I. LANGUAGE
COMPILE & DISPLAY
DEBUG LOOP
YES
EDIT ?
NO
TRIAL RUN ON SAMPLE STUDENTS
SELECT AND TRAIN TEACHERS

Overall Schematic of CAI System

Figure 3. 1

entire range of subject matter. Students are given a final examination which is drawn from the questions appearing on these tests. The achievement test problems and questions define the course content which is then to be taught. The function of curriculum writing is to take the subject matter, whatever it may be, and convert it into a form which students will assimilate.

The classroom teacher, standing before his students, must convert knowledge of subject matter into orally presented instructional material and assignments. The writer of programmed instructional material also converts the subject matter content into curriculum messages but of a very specialized type, called "frames." From the writing of programmed instruction a method of producing curriculum material for CAI has evolved. The people who produce the curriculum are called "authors."

The authors organize the subject matter into fairly broad "topics," and these are broken down into " concepts." The facts and elements of the concepts are called "teaching points." All of these specialized terms describe amounts of information in the same sense that chapters, sections, paragraphs and sentences are descriptive of the organization of the material in a book. There is very little true science to the arrangement of subject matter content in preparation for curriculum writing. There is also considerable controversy over exactly how the material should be presented to the student.

The authors, after outlining the material at the most detailed (teaching point) level, then must convert this material into instructional units called "frames." A frame consists of a stimulus, a response and a feedback portion. The stimulus is material containing the teaching point. The stimulus presumably is processed (read and understood) by the student and elicits a response from him. This response is processed by the machine. The student is then given feedback: information about the correctness of the response.

If a student provides the desired response to a stimulus, his feedback is a "reinforcement" (in psychological terms), that is, he is rewarded; usually by being told that he responded correctly. If he does not respond appropriately to the stimulus he is provided with "remediation," an instructional message designed to correct the student's misunderstanding. Desired responses are rewarded, undesired responses are corrected immediately. This feedback system is called contingent reinforcement.

Decisions on subject matter arrangement and presentation strategies are made by the authors who are usually organized into teams. Their job is to develop criterion tests and performance standards on tests, as well as to organize the material as described above. They are also responsible for coding the material in a form that is acceptable by the computer system. This involves a considerable amount of editing and debugging of the curriculum (see Figure 3. 1).

To perform these functions a group of curriculum writers generally includes persons conversant with: learning theory, the particular subject matter area, the parameters and constraints of the computer system they are using, and the particular CAI language in which their curriculum is written [5]. Some members of the team also frequently have classroom teaching experience.

The output of the curriculum writing group is an error free computer program. This program presents information to the student, both text and still and moving diagrams, processes his responses by classifying them in several ways, decides on what information to present next (given the student's previous responses) and gathers detailed information on the student's interaction with the curriculum. Curriculum programming required for one student console hour may cost between $600 and $1000 to program, debug and test.

The actual coding of the curriculum is enormously complex. The demands on the authors are largely determined by how many of the detailed functions of the computer system are left under their control. This is completely determined by the CAI programming language that is available.

2) Students and The Learning Process

The students in a Computer Assisted Instruction system are viewed as individual information processors. There are many schools of thought on how students process information. In very few cases it is possible to derive from these models of learning a scheme for presenting instructional material to the student. While the subject is a vast one and the available literature is not conclusive we will summarize and briefly discuss the more important views.

In describing students, it is difficult to distinguish clearly between teaching practice and learning theory. Even some noted authorities have made confusing statements by using highly abstract descriptions. Stolurow [8, p. 51] offers us an example in his discussion of alternate views of the student. Stolurow says that students may be viewed in two ways. These are: "(a) that the learner is a receptive mechanism for whom associative connections become formed so as to mirror experience; (b) that the learner is a selective self organizing mechanism who selects and extracts information from the environment." This statement is difficult to use as a guide to teaching or to student learning behavior.

Skinner's view One of the most cogent statements made about student learning has been made by Skinner [7]. He uses his contingencies of reinforcement as a vantage point from which to view three classical descriptions of the teaching-learning process. These three are (1) we learn by experience, (2) we learn by doing, and (3) we learn by trial and error. In Skinner's view these three descriptions of learning each emphasize only one of the contingencies of reinforcement (the occasion of a behavior, the behavior itself and the consequences of the behavior).

Thus, learning by experience is an emphasis on the occasions (experiences) surrounding a behavior. Learning by doing is an emphasis on the behavior itself. And finally, learning by trial and error concentrates on the consequences of behavior. By forceful arguments, backed up with massive empirical results, Skinner presses the issue and recommends the study of learning in a way that recognizes all three contingencies.

Other views Another difference of opinion about the student is over the student's reaction to remedial feedback. One school of thought is that a student should make the correct response as often as possible.

Being told that he has responded correctly is the activity, in the view of this school, which will most effectively lead the student to form the desired associations, that is, to learn most rapidly. Thus proper responding is made quite easy by leading the student to respond as desired. This view is almost strictly Skinnerian and is the one on which programmed instruction was originally based [1, 3, 4, 6].

The alternate view suggests a different approach to instruction. The student is provided with the opportunity to learn something, he is tested on it, and if he hasn't mastered it, the material is presented again. The presentation might be varied in some way in the hope that it will be more easily understood by the student. This approach has given rise to numerous variations on the original programmed instruction. The most notable of these is the Crowder or intrinsic program [2, 4]. In this type of program an incorrect response by the student is regarded only as information to be used in selecting the next piece of instruction.

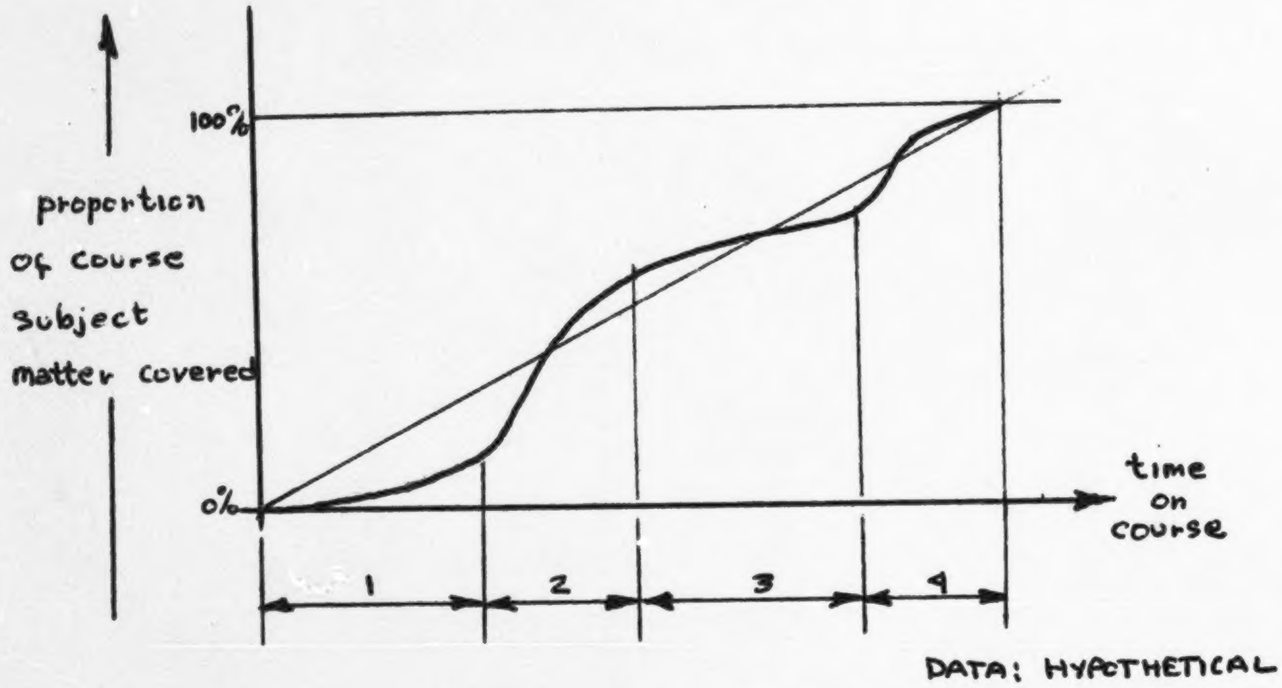
This second view of how the student should be remediated is the one which appears to be used by Project GROW.

Although we have said that the student is viewed as an "individual information processor," he obviously has other characteristics of motivation and attitudes which affect the way in which he learns (or does not learn). Researchers are constantly looking for measures of the effects of these affective characteristics, however, there are no widely acceptable or repeatable results yet.

The student and the curriculum interact via the computer and there are several points that involve both student and curriculum. One critical distinction is between the amount of time spent in study of the curriculum and the amount of curriculum exposed to the student. A further distinction which will be important later is the distinction between exposure to curriculum and the ability to answer questions on the final achievement examination.

Two factors cause a difference between the amount of time that a student spends on the curriculum and the relative amount of course content (measured by the number of teaching points) which the student has completed. The first of these is the difficulty or complexity of different subject matter topics. The second is the relative speed of the student in covering both instructional curriculum material and remediation which he might get because of wrong answers to various questions in the curriculum.

First, consider the effects of variability of difficulty in subject matter points. When the curriculum writers anticipate difficult topic areas (topics which will be difficult for the students to understand), they break the instructional messages into finer steps. They also make allowances for review because of the expanded treatment. Thus the student has more instructional material to process. Figure 3.2 illustrates this graphically. Proportion of course subject matter covered is defined as the proportion of course teaching points complete.



Progress on Subject Matter vs. Time on Course
Figure 3.2

In intervals 1 and 3 time spent on the course does not produce as rapid progress on subject matter as the times represented by intervals 2 and 4. The curriculum writers may have anticipated particularly complex material or perhaps the need for a large amount of motivational material in intervals 1 and 3. The expansion of the subject matter into added curriculum and then into extra time on the course means that the student's rate of progress through the subject matter is not truly uniform with respect to time.

The second factor in the relationship between time on the course and amount of subject matter content covered is dependent upon the student's speed of processing the curriculum messages. Here again there is a complicating factor.

When the student is presented with an instructional message (a frame consisting of stimulus-response and feedback), he should master the point being taught. Naturally this is often not the case. If a student has not learned a given item, he will make the wrong response to an item of curriculum, and the computer will branch him to an appropriate piece of remedial curriculum as his feedback (contingent reinforcement).

The amount of remediation which the student experiences will depend upon several factors; e. g., the number of points in the curriculum (or course) where a branch to remedial material is possible, the average number of frames of remedial material once a student starts remediation, the effectiveness of the original instructional material, and the student's ability to understand the initial instructional material.

As the percentage of correct responses made by the student goes down, the more remedial material he sees since remediation is contingent on wrong responses. The increased number of remedial frames decreases the effective number of frames of instruction that are relevant to the final examination in the course. That is, the effect of remediation is to increase the average time spent on each piece of subject matter content. (Where a student is particularly anxious to complete his computerized instruction, an interesting observation is that he often will try all the harder on the instructional material. If he makes careless or thoughtless responses he will have to be remediated and this will slow him down. Some students are quick to find this out!)

(3) Teachers

The third input to the instructional process of Figure 3.1 is the teacher. In the environment of the Philadelphia Project GROW the teacher is viewed as being the "tactical" manager of the CAI system. After student selection and scheduling decisions are made, the teacher monitors the student's interaction with the curriculum. Depending upon the teacher, the student's exposure to the computer may be changed if he goes too far ahead or lags too far behind any off-line instruction.

The teacher is also supposed to supplement the computerized curriculum with materials which may not be covered on-line. While the students are at the computer, the teacher's function is purely supervisory. If there are more students in the class than can be served by the consoles at one time, the teacher provides standard supplementary instruction to those who are waiting for consoles.

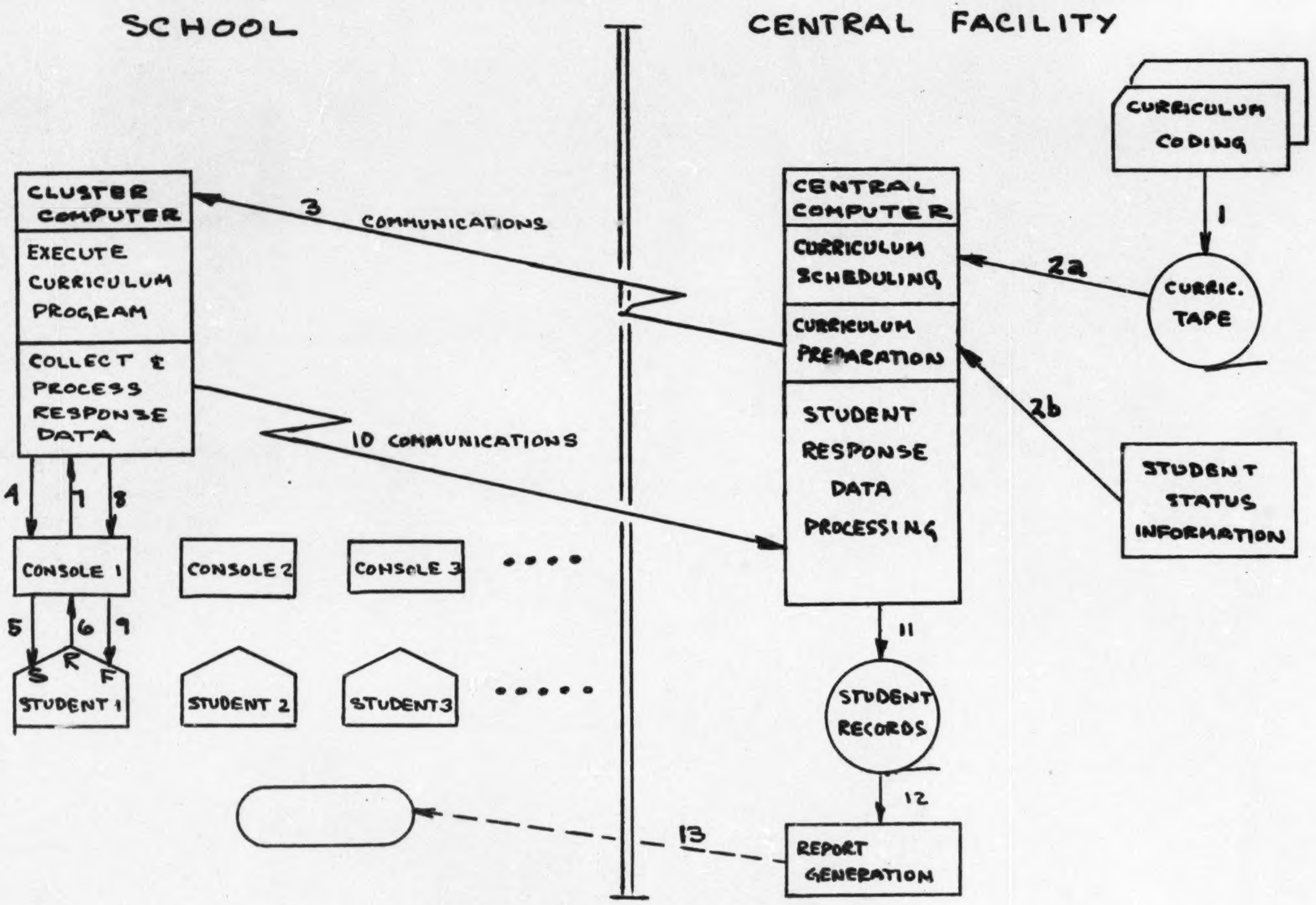
ORGANIZATION OF THE COMPUTER SYSTEM: STRUCTURE, COMMUNICATION, AND CONTROL

(a) Structure

Figure 3.3 describes the organization of the system. While the present Project GROW system consists of four schools, only one is shown and only one student is shown although eight consoles (a cluster) are operating at each of the schools. In describing the structure of the system we start with the hardware. There are two processors, one at the "central" and one in the school or "cluster."

The central unit selects the student's curriculum, given his progress to date, and transmits this information to the appropriate cluster. It also maintains student record and produces off-line reports of student progress. Essentially, the central computer performs coordination and control activities. Once it prepares the curriculum for a group of students its function is to receive and process information on the students' interaction with curriculum.

The cluster computer is the working end of the system in terms of processing the programs compiled from the curriculum writers' coding. It stores the curriculum received from central on a disc memory and when the scheduled student appears at a console it executes the proportion of the curriculum which pertains to that student.


SCHOOL
CENTRAL FACILITY
CURRICULUM CODING
1
CURRIC. TAPE
2a
2b
STUDENT STATUS INFORMATION
CENTRAL COMPUTER
CURRICULUM SCHEDULING
CURRICULUM PREPARATION
STUDENT RESPONSE DATA PROCESSING
11
STUDENT RECORDS
12
REPORT GENERATION
13
3
COMMUNICATIONS
10 COMMUNICATIONS
CLUSTER COMPUTER
EXECUTE CURRICULUM PROGRAM
COLLECT & PROCESS RESPONSE DATA
4
7
8
CONSOLE 1
CONSOLE 2
CONSOLE 3
5
6
9
S
R
F
STUDENT 1
STUDENT 2
STUDENT3

Information Flows in the CAI System

Figure 3.3

The cluster computer manages eight students simultaneously. At points designated by the curriculum writers, the cluster computer takes information which has been gathered on a student, formats it and transmits it back to central.

(b) Communication

Each of the numbered arrows shown on Figure 3.3 represents a flow of information or data. Some of the flows are man to machine such as the student's typing of a response on the keyboard of the console. Other flows are strictly between machines, for example, when the cluster computer transmits data on student performance back to central.

Beginning with the curriculum writers, key-punched curriculum program decks are converted to magnetic tape which can be accessed by the computer. This is represented by arrow number 1. The parts of the curriculum appropriate for the students scheduled to be on the system are selected (2a and 2b) by the central computer which transmits them to the appropriate cluster (3) at one of the four schools.

The student signs onto a console at the cluster computer and a most important exchange begins. First, the computer provides the student with an instructional message and calls upon him to make a response. This occurs by the transmission of the appropriate signals to the console (4). The message is displayed to the student (5) and the keyboard or light pen of the console is turned on. (The light pen enables the student to respond merely by pointing to a sensitized area of the screen.) The computer now waits for the student to process the stimulus message displayed and to enter his response (6) by keyboard or light pen depending upon how the authors wrote the original curriculum. While the computer waits it records the length of time required for the student to process the stimulus and to make his reply. After the student decides on his response, he enters it on the console (6). It is transmitted back to the cluster computer (7) where the programmed curriculum dictates the feedback (either reinforcement or remediation) which is to be given to the student (arrows 8 and 9).

Each of these stimulus-response-feedback interchanges is a frame. It is the basic unit of instructional communication. The length of time that the student takes between stimulus and response is defined as his response latency. Ideally, response latency is a measure of the student's mental processing speed. Actually, it is also dependent upon his reading ability and, where keyboard responses of more than a few symbols are involved, it may depend on his manual dexterity and typing ability as well.

After a number of these stimulus-response-feedback frames the cluster computer transmits the stored data on latency, frequency of correct responses, and other statistics back to the central (10). At the central the student records are processed and used to produce a data tape (11). The central computer is used during off peak time to produce reports from these tapes (12), and the reports are periodically sent to teachers (13).

We have sketched only the primary flows of information and data. With the data base established from these operations other reports are also generated. Examples of other types of communication would be the flow of information from operating data to the curriculum writing group. This closes the loop on curriculum production, and with performance data in hand the curriculum writers are able to improve the existing curriculum.

Appendix 3. 1 describes the data base that is provided by the system and that is used later in this thesis.

(c) Control

The discussion of control of the system will concentrate on those who are concerned with managing the system as an educational facility. The responsibility for making effective use of the system in educating students is not completely centralized. Varying degrees of control are lodged with the individual classroom teachers, their supervisors, the school principal, and with the overall system management.

The overall system management exercises control at the strategic level. The most crucial controllable elements which are reversable only at high cost are in the areas of system design (hardware and software).

The area of student selection and scheduling as well as educational performance evaluation is an area in which control is shared between the overall system management and the principal or school head. The performance of the whole system is directly connected with the numbers and types of students who interact with it. Selection and scheduling of students onto the system had not been recognized as an important controllable variable by those who can vary it.

Finally the detailed tactical control over student-computer interaction lies in the hands of the teachers. Aside from occasionally prescribing special work for students the teacher has direct control over the length of time which each student has at the console. Obviously, the need for the teacher to "fine tune" the students' schedule assignments is dependent on how widely their demands for time on the system vary. Students who work slowly will require more time while faster students require less. Unless the original scheduling system took this into consideration in building the original class assignments the differences can be large.

ASSUMPTIONS

In establishing the background for this research in the preceding sections many factors have been given very brief treatments. Essentially we have defined a very complex system. Later, a mathematical model of the interaction of a student with the curriculum is to be built. Since it cannot reflect all of the complexity of the system we state certain assumptions about the students and the curriculum.

Assumption 1 ... While a student is at the console, interacting with the curriculum, his cognitive (non attitudinal) learning is dependent only upon the curriculum displayed to him. The effects of off-line instruction are not included in the models.

Assumption 2 ... The course requires the student to engage in only one type of learning. We assume that the course does not mix learning of motor skills (e.g. typing or writing) with learning of verbal skills (e.g. reading or spelling.)

Assumption 3 ... The course curriculum is assumed to be a homogenous stream of instructional items which are all similar in terms of their difficulty and demands upon student attention.

Assumption 4 ... The probability that a student will make a wrong response on a frame that covers new course subject matter is constant for that student on that curriculum. (A student who has a high probability of wrong responses will make heavy demands for remediation.)

Assumption 5 ... Once the student has made a wrong response on a new instructional frame he is remediated. After remediation he can make the correct response to the instructional frame which he originally misunderstood. (That is, it is assumed a student misses a basic frame only once.)

Assumption 6 ... The latency of a student--the average time between instructional stimulus and the student's response--is a characteristic of that student which is constant over time and over the curriculum.

Assumption 7 ... A final examination of N questions is administered at the end of the course. The percentage correct score on this test is an estimate of the true proportion of course subject matter learned and retained up to the examination.

SUMMARY

The elements of a CAI system: (1) course design and curriculum, (2) students, (3) teachers, and (4) computer system, have been defined. The basic element of course material, the instructional frame, causes the student to make a response to an instructional stimulus. The computer uses logic, provided by the course authors, to evaluate this response and to provide instant feedback to the student. The detailed flows of curriculum and response data through the computer system have been described. Assumptions about the course curriculum, about the student and his learning behavior, and about the final test on the course have been stated.

APPENDIX 3.1

CHAPTER THREE

SYSTEM DATA SOURCES--ON LINE DATA

In later sections of this thesis we present results of empirical studies and tests of models, largely based upon data that the computer system gathered during course operation. This appendix describes the system data sources used in the empirical work. The purpose here is to allow other researchers to repeat or extend analyses reported in this thesis. Data gathered from sources other than the Philco-Ford computer system will be referred to as off-line data. Off-line data sources are identified when they are used and will not be discussed in this appendix.

The main source of system data used in empirical work was the TOPIC SUMMARY TAPE described in the Project GROW-File Description memorandum dated 9/28/67. The diagram below (reproduced from that memorandum) describes the format of the records on this tape. Each record on the tape contains the data gathered during one topic of curriculum; that is data resulting from a student's interaction with curriculum displayed in the time between the execution of a DEFT command and the next ENDT command of the INFORM curriculum source program. The fields of the records are named.

<table>
<tr><td>e</td><td>U</td><td>c</td><td colspan="2">T</td><td colspan="3"></td><td colspan="4">N</td></tr>
<tr><td colspan="2">N</td><td>C</td><td>S</td><td colspan="2">CC</td><td colspan="2">SN</td><td colspan="2">SF</td><td colspan="2">TT</td></tr>
<tr><td colspan="12">TN</td></tr>
<tr><td>RC</td><td>UT</td><td colspan="4">For Expansion</td><td colspan="2">TC</td><td>SQ</td><td>RA</td><td>C1</td><td>C17</td></tr>
<tr><td>C2</td><td>C18</td><td>C3</td><td>C19</td><td>C4</td><td>C20</td><td>C5</td><td>C21</td><td>C6</td><td>C22</td><td>C7</td><td>C23</td></tr>
<tr><td>C8</td><td>C24</td><td>C9</td><td>C25</td><td>C10</td><td>C26</td><td>C11</td><td>C27</td><td>C12</td><td>C28</td><td>C13</td><td>C29</td></tr>
<tr><td>C14</td><td>C30</td><td>C15</td><td>C31</td><td>C16</td><td>C32</td><td colspan="2">SW</td><td>R1</td><td>R5</td><td>R2</td><td>R6</td></tr>
<tr><td>R3</td><td>R7</td><td>R4</td><td>R8</td><td>LRL</td><td>ARL</td><td>CAC</td><td>WAC</td><td>AAC</td><td>UAC</td><td>TAC</td><td>XAC</td></tr>
</table>

(A) = ASCII

e = Hex. 70 (A)
U = User ID (A)
c = Course ID (A)
T = Teacher ID(A)
N = User Name (A)
C = Cluster ID (A)
S = Schedule (B)
CC = Current Concept (A)
SN = Last Sign On (B)
SF = Last Sign Off (B)
TT = Time on Topic (B)
TN = Topic Name (A)
RC = Response Counter (B)
UT = User Type
TC = Time on Course

(B) = BINARY

SQ = Seq. No. of Asso. Response Record (B)
RA = Relative address of restart Pt. (B)
C1 - C32 = Programmable counters (B)
R1 - R8 = Return Registers (B)
SW = Switches (B)
LRL = Topic Last Response Latency (B)
ARL = Topic Average Response Latency (B)
CAC = Topic Correct Answer Counter (B)
WAC = Topic Wrong Answer Counter (B)
AAC = Topic Anticipated Answer Counter (B)
UAC = Topic Unrecognized Answer Counter (B)
TAC = Topic Time Up Answer Counter (B)
XAC = Topic X Inclusive Answer Counter (B)

Description of the Record Format on the

TOPIC SUMMARY TAPE

The tape as received from the School District coding group was translated from ASCII character representation to EBCDIC representation by means of a specially written PL/1 program. This step was necessary to allow analysis on the University of Pennsylvania's IBM 360/65. Because of the large volumes of data and the processing involved in translating the tapes, a small sample of records were checked by manually comparing hard copy printouts of the PHILCO-FORD records with translated records. All further analyses of system data were carried out on these translated tapes.

DATA ELEMENTS USED

The data elements which have been used in this thesis are the system counters LRL, ARL, CAC, WAC, AAC, TUC, UAC, XAC and TT. The values in these counters together with information on the user identification and location on the curriculum provide the raw data for analysis. In some cases--particularly in the latency counters ARL and LRL--there were obviously faulty values. These errors were due to malfunctions of the hardware clock which is the source of timing information for the record keeping routines. Whenever calculations were performed on the latency data, each value was tested for reasonableness. Negative values and values of greater than 1000 seconds were replaced by that particular student's overall average latency. These errors did not occur frequently; less than five percent of the data was edited in this way. The cause of these discrepancies in the data gathering facility is being eliminated.

REFERENCES

Chapter Three

[1] Becker, James L. A Programmed Guide to Writing Auto Instructional Programs. Camden, N. J.: R. C. A. Service Co., 1963.

[2] Crowder, Norman A. "Simple Ways to Use the Student Response for Program Control." In Applied Programmed Instruction, Margulies/Eigen (ed.), New York: John Wiley & Sons, 1962.

[3] Hilgard, Ernest R. Theories of Learning. New York: Appleton Century Crofts, second edition, 1956.

[4] Hughes, J. L. Programmed Instruction for Schools and Industry. Chicago: Science Research Associate, 1962.

[5] Silvern, Gloria M., and Silvern, Leonard C. "Computer Assisted Instruction: Specification of Attributes for CAI Programs and Programmers." 1966, Proceedings of 21st National A. C. M. Conference, pp. 57-62.

[6] Skinner, B. F. "Teaching Machines." In Teaching Machines and Programmed Learning, Lumsdaine and Glaser (ed.), Washington, D. C.: Department of Audio-Visual Instruction, National Education Association, 1960.

[7] ______. The Technology of Teaching. New York: Appleton Century Crofts, Division of Meredith Corporation, 1968.

[8] Stolurow, Lawrence M. Teaching by Machine. Washington, D. C.: U. S. O. E. Monograph No. 6, U. S. Government Printing Office, 1961.

CHAPTER FOUR

A MODEL OF STUDENT--CURRICULUM INTERACTION

INTRODUCTION

Essential to any procedure for allocating CAI use is a model relating each student's mastery of subject matter to his time on the CAI system. This learning model must represent several characteristics of the student and of the computerized curriculum. Both are enormously complex, so it would be hopeless to try to include in the learning model every nuance of student and curriculum. In developing a model, then, the two considerations that will apply are parsimony and usefulness. The usefulness of a model is judged by its ability to explain a student's learning behavior in terms of his interaction with the computerized curriculum [1, 5].

BASIC ELEMENTS OF THE LEARNING MODEL

Index of Mastery

In speaking of mastery of subject matter, we are interested in concrete evidence of learning. The performance gauge will be an achievement test on all subject matter covered in the course. The test consists of a representative sample of N problems and questions selected from the original tests which the authors used in designing the course. When it is necessary to test a student before the end of the course, one of the original topic tests will be used. The score on this test is made comparable with a score on the final N-item test in this way: let the number of questions on the topic test be I, let the number of questions on the final test be N, and let the score on the topic test be s; then the adjusted topic test score is $x=s(I/N)$.

Consider a typical student who has finished a total of T time units of instruction. From time to time we interrupt his instruction and give him an interim achievement test over the topics covered so far. When these scores are converted to a comparable basis, the student's progress through the course can be seen by looking at the time series $\{x(t)\}$ of test scores,

$$x(t_1),\ x(t_2),\ x(t_3), --- x(T).$$

In other words, if a student has interacted with the curriculum for t time units, an estimate of the score he would attain on the final achievement test is $x(t)$.

Factors Affecting Mastery of Subject Matter

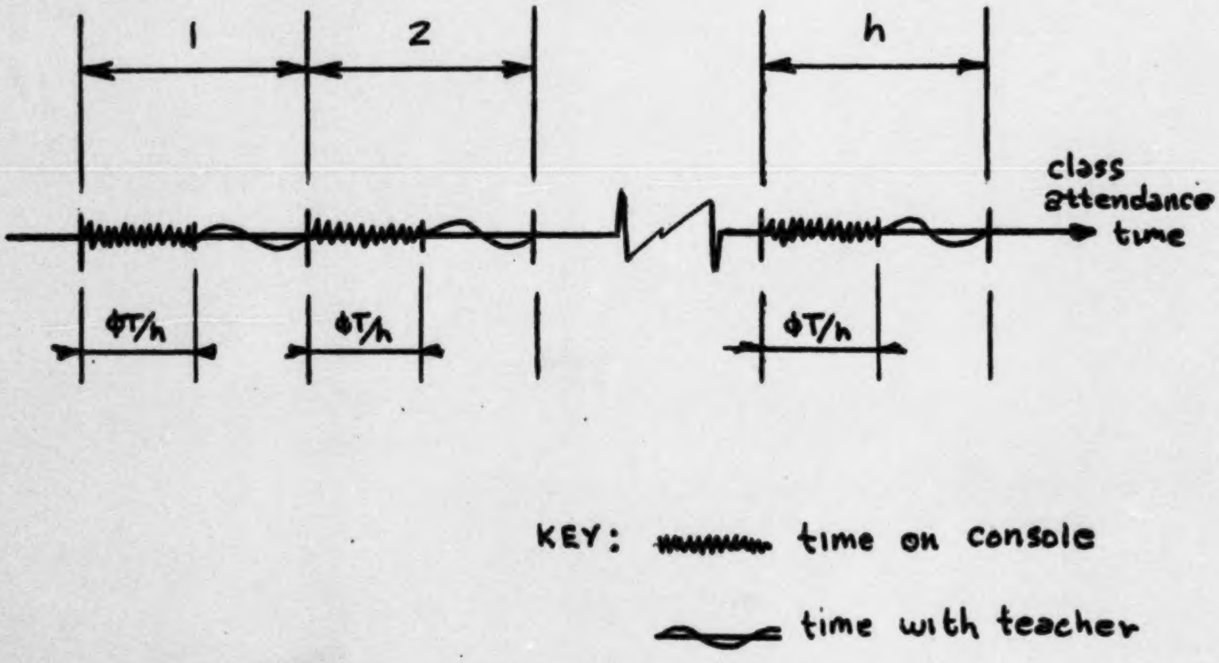
Two factors that affect ultimate learning performance are pace of work and ability to retain information presented by interaction with the curriculum. Numerous underlying factors come together to influence and determine work pace and retention. The student forms association between cues and desired responses by reading and responding to instructional messages of the curriculum. How quickly and effectively this interaction takes place depends upon the quality of the instructional and remedial instructional program, student motivation and interest in the course, "intelligence", attentiveness, impulsiveness, desire to complete the curriculum and to learn the subject matter, and reading speed and comprehension. The effect of the variation of these factors from student to student will be different values of parameters in the learning models. For an individual the parameter values will be assumed constant and typical of him--for the curriculum used.

STUDENT MODEL I

The effort to design a learning model ultimately led to one which is supported by available data on student-curriculum interaction. A description of the first attempt to model this interaction is presented to trace the history and development which led up to Model II, the version which will be used in the allocation procedure. The first model will not be used later since it was not supported by available data.

Growth of Amount of Material Learned

The starting point in the process of developing an acceptable learning model was a system of finite difference equations. Starting with a total of T units of student attendance time, we assume that this can be broken into a number (h) of equal periods. In each of the individual periods, (T/h) time units long, we suppose that the student is scheduled onto the console for $\Phi T/h$ time units; Φ is the proportion of the student's time on the console. A proportion $(1-\Phi)$ of the student's time is spent with the teacher. The figure below illustrates the pattern of instruction schematically.


1
2
h
class attendance time
ΦT/h
ΦT/h
ΦT/h
KEY:
time on console
time with teacher

Pattern of Instruction

Figure 4.1

We assume that the student is given a test at the end of each of the periods 1, 2, ---, h. These scores

$$x(T/h),\ x(2T/h),\ x(3T/h), ---, x(T)$$

will be written $x_1, x_2, ---, x_h$ for simplicity. We will also use the symbol x_0 for the fraction of course subject matter which the student knows before he starts the course.

By assumption 7, x_0 is the proportion of questions that the student can correctly answer on a 'final' exam just before his course attendance time starts. By the time one period of length T/h has passed and in the absence of any instruction, the student will have forgotten some information which he had known earlier. Assume that the proportion of course subject matter learned will drop by a fixed amount. If there is no instruction during the first period $[0, T/h]$ we write:

$$x_1 = \alpha x_0 \qquad 0 \leq \alpha \tag{1}$$

Here, α is the proportion of learned material which the student retains from one period to the next. If there is no instruction we have $0 \leq \alpha \leq 1$. (It is conceivable that a student who studies course material outside of school could appear to have $\alpha > 1$.) The main way in which we increase the student's knowledge of course subject matter, however, is to teach him.

During the period $[0, T/h]$ the student is exposed to the computerized curriculum for a period of $\Phi T/h$. The remainder of the period, $(1-\Phi) T/h$, is spent with the teacher. The effect of computerized instruction is to increase the student's test score in some way. We represent the effects of instruction during the first period by the symbol $\Delta x_1(\Phi)$ and change equation (1) to include the effects of instruction. The nature of the quantity, or unit of instruction, Δx_1 is discussed in the next section. Now we rewrite equation (1) to reflect the contribution of the first period of instruction. Thus

$$x_1 = \alpha x_0 + \beta \Delta x_1 \qquad \begin{array}{l} 0 \le \alpha \le 1 \\ \beta \text{ constant} \end{array} \qquad (2)$$

β is a parameter which would be estimated for a particular student and a particular period of instruction. A theoretical interpretation of the β would require it to represent two effects. First β reflects the weight of that period's instructional material relative to the whole course's subject matter. Second, β serves the purpose of a scale factor. The quantity Δx represents the effect of a relatively short period of instruction. The quantities x_0 and x_1 represent the overall proportions of course subject matter which the student has mastered before and after the first period. The constant β can be interpreted as the relative efficiency with which the student converts the results of recent instruction into long term subject matter knowledge. The fact that human memory has short term and long term storage capability can be used to propose an interpretation of β [2, 4]. β is an indication of the efficiency with which recent instruction (presumably retained in short term memory) is transfered into long term memory associations.

The final justification for the parameters is not only the interpretations which are assigned to them, but the validity of the resulting model when tested empirically.

So far only the first period of instruction has been discussed in equation (2). Other periods of instruction must be represented in the model and this is accomplished by assuming that α and β are constant characteristics of the student. The general relation, which includes equation (2) as a special case, is then:

$$x_i = \alpha x_{i-1} + \beta \Delta x_i \qquad i=1,2,---,h \qquad (3)$$

This system of equation can be used to find the final test score x_h in terms of all previous instruction and x_0.

$$x_h = \alpha^h x_0 + \sum_{i=1}^{h} \alpha^{h-i} \beta \Delta x_i \qquad (4)$$

In the special case where all Δx_i are equal, say to some value Δx we may rewrite equation (4) as:

$$x_h = \alpha^h x_0 + \frac{1-\alpha^h}{1-\alpha} \beta \Delta x \qquad (5)$$

A Unit of Instruction

In programmed instruction and CAI a student learns by processing a sequence of instructional messages (frames) which the teaching medium presents to him. It is not clear that the concept of a frame is useful in describing instruction provided by a human teacher. On the other hand, there is no conclusive evidence to rule out the use of frames as a description of the classroom teacher's interaction with individual students. The development below assumes the frame concept is meaningful in classroom teaching.

The quantity Δx_i, used in equation (4) will be the effective number of instructional frames, covering new course subject matter, during instructional period i. This includes both the frames explicitly presented by computerized instruction and the estimated number of frames due to the teacher's efforts. The effective number of frames processed by the student during the i-th period of instruction (Δx_i) will be related to several factors. The list of symbols below will be required in developing the relationship which defines Δx_i.

λ^c : Student's rate of processing new instructional frames on the computer [frames/time].

λ^t : Rate of delivery of frames by teacher.

p^c : Probability that the student will make the desired response to a frame presented by the computer; called the interaction probability. It is by definition the probability of reinforcement on a given frame.

p^t : Probability that the student will have his response reinforced on a randomly selected frame presented by the teacher; called the interaction probability of the student with the teacher in the classroom.

The effective number of frames during a period is defined to be the sum of the numbers of frames presented by both the computer and the teacher that the student interacts with. The student is assigned to be on the console of $\Phi T/h$ units of time. During this time he will average $\lambda^c \Phi T/h$ frames and he will interact with a fraction p^c of these. While with the teacher the student experiences $\lambda^t(1-\Phi)\ T/h$ frames, and he interacts with p^t of these. Assuming that the teacher covers material not covered on the computer, we write the effective number of frames as:

$$\Delta x_i = \lambda^c p^c \Phi\ T/h + \lambda^t p^t (1-\Phi)\ T/h \tag{6}$$

In the second term of equation (6), $\lambda^t p^t (1-\Phi)\ T/h$, p^t allows for the fact that the student may not process all of the messages presented by the teacher because of his attentiveness or interest; λ^t allows for the fact that the teacher presents curriculum material at a characteristic rate.

The Learning Model

At this point we are ready to substitute equation (6) into (5). The resulting function relates the student's final test score to the proportion of his time on the console, Φ, and to his individual learning characteristics. The final relationship is

$$x_h = \alpha^h x_0 + \frac{1-\alpha^h}{1-\alpha}\ \frac{\beta T}{h}\ [\lambda^c p^c \Phi + \lambda^t p^t (1-\Phi)] \tag{7}$$

This relationship will be referred to as Model I.

An attempt was made to validate a version of this model on the computerized portion of a course in biology. Instructional histories of five students from the Spring 1968 biology class at Germantown High

School were used to fit the parameters of the model by least-squares regression. The model did not stand up as a statistically defendable description of the available data.

The failure of the model may have been just as much due to insufficient data as to any fault of the model itself. Model I assumes that a test score is available at the end of each period of instruction. In fact, tests were not given this frequently, and the resulting missing data problems severely hindered tests of the model. Several statistical procedures for eliminating the missing data problem did not produce satisfactory results.

Because of these difficulties, and after finding that it was not possible to gather longer histories on the students, a model was developed that is much less sensitive to the placement of tests within the course of instruction. This model, Model II, is developed below.

STUDENT MODEL II

Model I attempts to provide a rather detailed description of the interaction between the student and the instructional process. The model makes heavy demands on the researcher for data to be used in estimating parameters. The second model represents a step in the direction of parsimony. This model deals with aggregate results of the learning processes of the student; it is based upon the cumulative effects of the teaching-learning process.

As in the last section x(t) will represent the student's score on a final achievement test administered at time t. When the student's total time on the curriculum is small (t is small) we shall assume that his test score changes in proportion to how much he has learned up to that point. The dominant factors in initially changing the student's test score are his ability to read and attend to the curriculum material as it is displayed, to make associations between old and new material, and to retain it as a whole. For this aggregate model, understanding the details of this learning process is not essential.

As the student progresses through the course, that is, as the student learns more and x(t) grows, there are several factors which act to impede the rate at which his knowledge increases further. His interest in the course may lag, he will tend to forget some of the material learned earlier, and he will encounter some of the more difficult areas of the subject matter. These factors--forgetting, and a tendency toward a certain amount of confusion--have an opposite effect from that of learning. These effects are all represented in the way that the student's score on the final achievement test changes over time. Now we translate these statements into a mathematical model.

The symbols which we use in this model are:

t : Time.

$s(t)$: Raw score on the final achievement test if it were given after the student has spent a total time of t units on the course.

N : Number of questions on final achievement test on computerized curriculum.

$x(t)$: The fraction of final achievement test questions scored correct after the student has spent a total of t time units on the course.

λ : Average rate at which the student progresses over instructional and remedial frames. (Frames/unit time).

p^c : Probability that the student interacts successfully with an instructional frame.

p : Proportion of time interacting with instructional frames.

c : Progress in units of curriculum.

The test score, s, will be a function of the amount of instructional curriculum material which the student has completed. As time goes on the student working at his own rate λ (frames per unit time) covers λt frames in a period of t time units. Out of the frames that the student sees a proportion are applicable to the final test. This proportion is the fraction of all frames on which (1) the student either responded correctly the first time, or (2) on which he first responded incorrectly and then was remediated. Thus the fraction of time which is applicable to the final achievement test (denoted by p) is $[p^c + (1-p^c)] / [p^c + 2(1-p^c)]$. In the period of time t the student has actually seen $\lambda(2-p^c)^{-1}t$ frames that apply to the final test. In developing the model we will work in terms of the cumulative curriculum covered, c,

$$c = \lambda(2-p^c)^{-1}t \equiv \lambda pt \qquad (8)$$

The scale will be changed back to real time after the derivation.

One of the consequences of equation (8) is that we assume that it takes just as long to remediate the student (given that he had made an incorrect response to an item of instruction) as it usually takes to instruct him on any point, without remediation. That is, the student is spending a proportion (1-p) of his time in remediation.

Early in the course, while t is small and c is close to zero we have that

$$\frac{ds(c)}{dc} \sim k_1 s(c) \qquad ; \; k_1 > 0 \quad \text{constant.}$$

The change in the student's knowledge of course subject matter is proportional to his existing knowledge of the subject matter. Later as his knowledge of course subject matter approaches an upper bound; as $s(c) \to N$ we have

$$\frac{ds(c)}{dc} \to 0 \text{ as } s(c) \to N$$

this asymptotic behavior can be modeled by:

$$\frac{ds(c)}{dc} \sim \left(1 - \frac{s(c)}{N}\right)$$

We can combine these expressions, and represent our previous statements about student learning in equation (9)

$$\frac{ds(c)}{dc} = k_1 s(c) \left[1 - \frac{s(c)}{N} \right] ; \; k_1 > 0, \text{ constant} \qquad (9)$$

This equation can be solved directly for s(c). After integration we obtain the results;

$$-\ln \left(\frac{N}{s} - 1 \right) = k_1 c + k_2 \; ; \quad k_2 \text{ constant} \qquad (10)$$

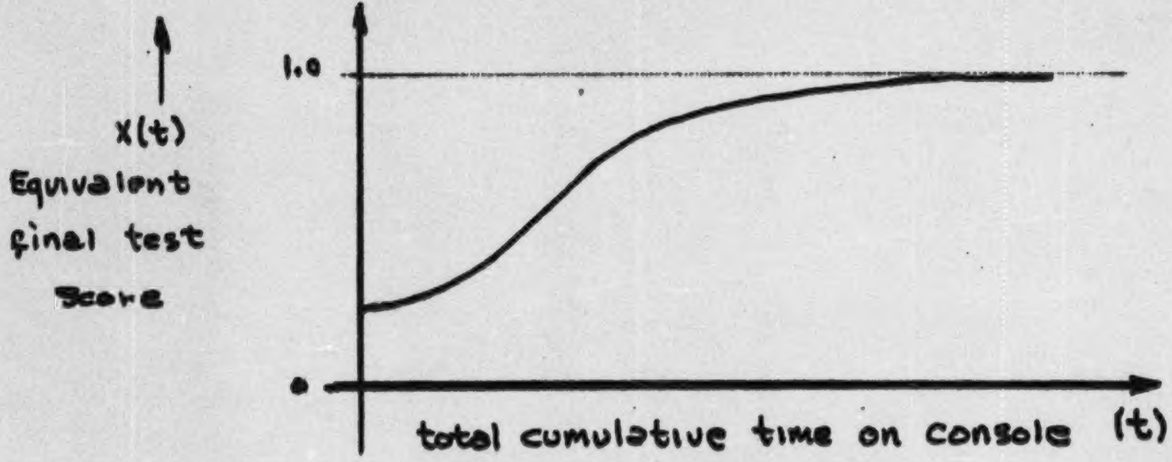
If we substitute equation (8) back into (10), we have an expression for the test score as a function of total time on the console, t,

$$-\ln \left(\frac{N}{s} - 1 \right) = k_1 \lambda p t + k_2 \qquad (11a)$$

By exponentiating both sides and using the fact that $x \equiv {}^{s}/N$ we have also

$$x(t) = \frac{e^{k_1 \lambda p t + k_2}}{1 + e^{k_1 \lambda p t + k_2}} \qquad (11b)$$

Figure 4.2 below shows the general form of the function (11b).



General Form of Learning Curve

Figure 4.2

Equation (11b) indicates that the value of x at the start, designated x(0), is

$$x(0) = \frac{e^{k_2}}{1 + e^{k_2}} \quad \text{if } t = 0 \tag{12}$$

The constant k_2 in this model depends only upon the student's knowledge of subject matter before the course starts. This fact is also obvious from (11a), a logarithmic transformation of the original nonlinear expression for x(t).

The coefficient of t in equation (11a) is $k_1\lambda p$. In the preceding chapter we assumed that the rate at which the student proceeds through curriculum material and that his requirements for remediation are both characteristics of the student which are constant at least for the curriculum. These assumptions now come into play. By applying them both λ and p are constants for a particular student over the duration of the instruction. As constants, for a particular student, both λ and p can be combined with k_1 to form a new constant, k_1', which is characteristic for that student over the course of instruction. The expressions for x(t) are rewritten as

$$-\ln\left(\frac{1}{x} - 1\right) = k_1't + k_2 \tag{13a}$$

and
$$x(t) = \frac{e^{k_1't + k_2}}{1 + e^{k_1't + k_2}}$$

so, $$x(t) = [1+e^{-k_1't-k_2}]^{-1}. \qquad (13b)$$

In evaluating the parameters k_1' and k_2, equation (13a) is the easier to use. By using the quantity $\ln(\frac{1}{x} - 1)$ rather than the test score proportion correct we have a linear function in terms of total time on the console. The standard approach of estimation by least squares can be used to provide an estimate of the values for k_1' and k_2. These estimates, however, may not result in a least squares fit of the non-linear function (13b) to the data.

ESTIMATING PARAMETERS FOR LEARNING MODEL II

The data that the computer system gathered on the computerized portion of the biology curriculum enabled testing of the models of student-curriculum interaction. It is impossible to gather data on the classroom portion of instruction which is as accurate and as complete as that gathered by the computer system. Thus, testing of the models has been restricted to the computer assisted part of the course. The teacher's effect on the instructional process is analyzed further in Chapter Six.

Statistical Analysis--Model II

Model II is a logistic curve of the type which has long been fit to learning data by experimental psychologists [3]. The model can be transformed from its direct form (equations 4, 11b) to an expression which is linear with a non-linear function of the original dependent variable on the left, (equations 4, 11a). Here x is the percentage final test score and t is total cumulative time on the console in minutes.

$$-\ln\left(\frac{1}{x} - 1\right) = k_1't + k_2 \qquad \text{for one student}$$

This model was fit to data on 20 students all of whom took biology at Germantown High School during Spring 1968. The data preparation is discussed in detail later in Appendix 4. 1.

Discussion of Results

The results of statistical analyses of the data from 20 students are summarized in Tables 4. 1 and 4. 2 below. In order to better show the student to student variability in Model II parameter estimates, the paired values of k_1' and k_2 are plotted on a scatter diagram (Figure 4. 3). The approximate extent of one standard error for each of the two estimates, k_1' and k_2, have been shown for two typical cases (cases 1 and 5).

In every case (Table 4. 1) the model's fit to the data is so close that there is only a remote possibility that the agreement could have resulted by chance. Furthermore, the model fit the data for every student on whom there was enough data available to test it (20 cases). And there is virtually no possibility at all that this event could occur by chance. A point of interest in Figure 4. 3 is that the paired

Student Number	Intercept k_2	t	Slope k_1'	t	r^2 adjusted for d. f.	Regression F	degrees of freedom F	degrees of freedom t
1	−1.9207	− 7.991	0.0078	7.077	0.9076	50.08	(1,4)	5
2	−1.3861	− 5.668	0.0073	5.2782	0.8174	27.86	(1,5)	6
3	−1.2512	− 7.3002	0.0059	8.3003	0.9314	68.89	(1,4)	5
4	−1.7170	−25.6452	0.0046	21.9189	0.9897	480.44	(1,4)	5
5	−1.4694	−12.082	0.0051	10.6819	0.9411	112.76	(1,6)	7
6	−3.1435	−14.9701	0.0142	11.1644	0.9537	124.64	(1,5)	6
7	−2.6116	−15.3688	0.0141	11.7421	0.9580	137.88	(1,5)	6
8	−2.1008	− 9.8721	0.0092	11.4139	0.9773	130.28**	(1,2)	3
9	−1.4708	−10.8288	0.0108	8.8758	0.9396	78.78	(1,4)	5
10	−3.0816	−10.0133	0.0132	8.7036	0.9373	75.75	(1,4)	5
11	−0.6364	−27.4593	0.0030	14.4407	0.9858	208.53	(1,2)	3
12	−3.3612	−20.0075	0.0186	12.2413	0.9738	149.85	(1,3)	4
13	−3.2170	− 9.9884	0.0143	7.3032	0.9280	53.33**	(1,3)	4
14	−2.9041	− 8.4273	0.0079	6.3696	0.8878	40.57	(1,4)	5
15	−3.1355	−10.5039	0.0088	7.4658	0.9012	55.74	(1,5)	6
16	−2.3408	−22.8975	0.0143	21.8668	0.9917	478.16	(1,3)	4
17	−3.8478	− 7.1569	0.0368	4.8145	0.8809	23.18*	(1,2)	3
18	−2.6289	−10.8752	0.0099	6.5117	0.8734	42.40	(1,5)	6
19	−2.2805	−17.0535	0.0083	11.6030	0.9570	134.63	(1,5)	6
20	−2.3637	−24.5039	0.0136	21.5232	0.9914	463.25	(1,3)	4

Note: All values significant at 99.5% level except
** : 99%
* : 95%

MODEL $-\ln[(1/(\text{test score}))-1] = k_2 + k_1' x$ (time on console in minutes)$\cdot(10-^{1})$

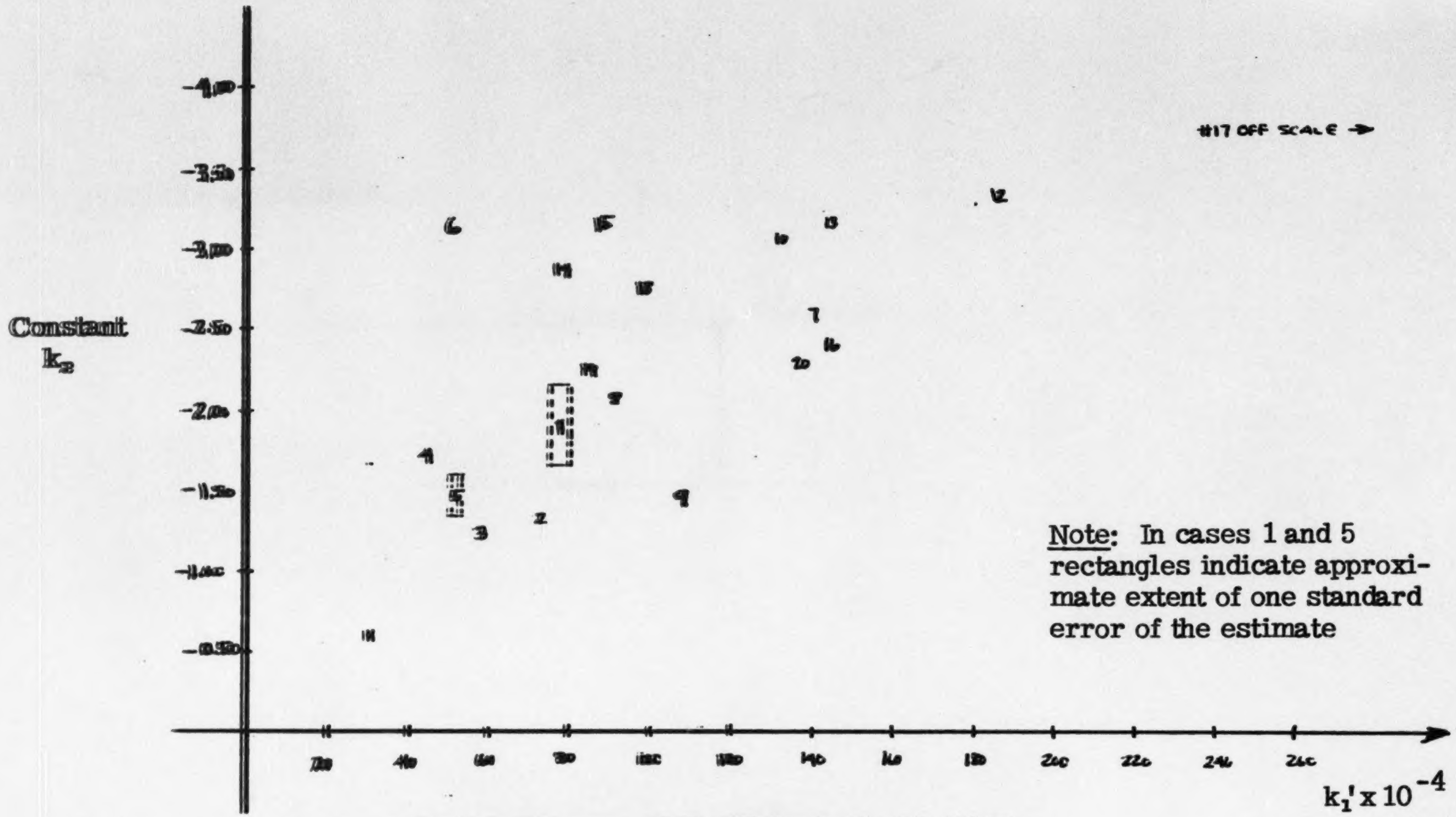
Point Estimates of k_1' and k_2 with Significance Tests

Table 4.1

Student Number	Intercept k_2 (Repeated from Table	Standard Error	Slope k_1' (Repeated from Table	Standard Error
1	-1.9207	0.2404	0.0078	0.001099
2	-1.3861	0.2446	0.0073	0.001373
3	-1.2512	0.1714	0.0059	0.000712
4	-1.7170	0.06695	0.0046	0.0002113
5	-1.4694	0.1216	0.0051	0.000478
6	-3.1435	0.2100	0.0142	0.001272
7	-2.6116	0.1699	0.0141	0.001997
8	-2.1008	0.2128	0.0092	0.000807
9	-1.4708	0.1358	0.0108	0.001218
10	-3.0816	0.3078	0.0132	0.001513
11	-0.6364	0.2318	0.0030	0.000209
12	-3.3612	0.1680	0.0186	0.001520
13	-3.2170	0.3220	0.0143	0.001962
14	-2.9041	0.3446	0.0079	0.00124
15	-3.1355	0.2985	0.0088	0.001184
16	-2.3408	0.1022	0.0143	0.000654
17	-3.8478	0.5376	0.0368	0.007637
18	-2.6289	0.2417	0.0099	0.00153
19	-2.2805	0.1337	0.0083	0.000714
20	-2.3637	0.0965	0.0136	0.000634

Point Estimates of k_1' and k_2 with Standard Errors

Table 4.2



Note: In cases 1 and 5 rectangles indicate approximate extent of one standard error of the estimate

Scatter Diagram of Coefficients for Model II

Figure 4. 3

values of k_1' and k_2 appear to be correlated. The apparent correlation was tested by regression analysis and it was significant at the 0.95 level ($r^2 = 0.5776$). This correlation was a surprise initially, but an explanation follows from Model II. In deriving the model it is assumed that during the early stages of instruction the rate of change of expected final test scores would be proportional to the level of test scores achieved. Thus, we would naturally expect a correlation between the slope k_1' (the initial rate of learning) and k_2 (the knowledge at the start.)

The data presented in Figure 4.3 show increasing values of the slope k_1' associated with smaller (larger negative) values of k_2. The explanation for this phenomenon lies in the fact that for small values of x very large changes in $-\ln\left(\frac{1}{x} - 1\right)$ must be made for x to change by even the small amount expected by the model. As an example suppose that a student scores 0.05 on his first test (over all course subject matter). Then after 100 minutes of instruction he takes a second examination and he scores 0.10 (over all course subject matter). The change in x by an amount 0.05 (0.10 - 0.05) is equivalent in the linearized model to a change of 0.64722.

$$\ln\left(\frac{1}{0.05} - 1\right) - \ln\left(\frac{1}{0.10} - 1\right) = 0.64722$$

With this change occurring over 100 minutes the slope k_1' would be

0.64722/100 = 0.00647. The smaller the initial test score the larger the slope k_1' will be in Model II.

These differences in model parameters from student to student are very important. Assuming that some way is found to estimate these parameters we can build the schedules of instruction for each student that will recognize that student's own unique requirements for time on the consoles. Chapter Six discusses the use of off-line data to estimate parameters in Model II on a student by student basis.

Conclusion

The two models which we have built are different descriptions of the way in which a student's knowledge of a course subject matter develop. One (Model I) is a detailed representation of the way in which a mixture of computerized and classroom instruction produce a growth of mastery. Model II is a more parsimonious description of the learning process. The two models are not isolated instances. The results of testing Model I prompted a move toward a simpler, more parsimonious model, Model II.

SUMMARY

A model of the relationship between time on an instructional system and a student's subject matter knowledge has been cited as the prime requirement for an instructional time allocation procedure. Two models have been proposed. Each relates an achievement test score to the preceding instruction. Model I was proposed to include the effects of teacher instruction as well as that of the CAI system. Model II is a simpler model of learning; and it has been tested with data obtained from the CAI system.

APPENDIX 4.1

CHAPTER FOUR

DATA DEFINITIONS

The data for the analyses reported above were taken from the TOPSUM Tapes supplied by the School District of Philadelphia--Instructional Systems Department (please see Appendix 3.1 for symbol definitions and description). The information supplied was from the first biology course ever to be taught using computerized instruction in an existing urban high school, Germantown High School, in this case. Some data was available on 31 students but there were only 20 students for whom there were sufficiently long histories to permit any analysis.

The original group of students taking the course was chosen by random selection from students who were scheduled for biology during spring 1968. Due to high student turnover rates and course switching some departure from the random sample occurred; in fact this is one of the reasons why several student histories were too short for analysis. A further check was made on off-line records for the 31 students. The 11 with short histories had a significantly higher rate of absences (during the preceding school year) than the 20 for whom enough data was gathered to estimate the model parameters.

Data Preparation--Model II requires a time series of test scores for each student. The elements in this time series must be on a comparable basis; in this case they are all equivalent values of the final course achievement score. Each test score has an associated time value. For a particular test score x(t) the associated time value was computed by summing the contents of the TT fields of all previous instructional topic summary records. The console time spent taking tests was not included in calculating cumulative time.

The tests which were given periodically to the students by the computer did not cover the entire contents of the course. Scores on tests that covered only part of the course were adjusted to reflect the proportion of the course material that the test had covered. The calculation of the values x(t) was carried out in the following fashion.

Each student file was checked to determine how many tests he had taken. The students whose files contained four or more tests were selected for analysis. Let r be the number of tests in a student file ($r \geq 4$). The number of questions on each of the r tests will be represented by $q_i(i=1,---,r)$; the number of those questions which the student answered correctly will be represented by $s_i(i=1,---,r)$, s for score. In standard mnemonics (please see Appendix 3.1) $s_i = CAC_i$; $q_i = CAC_i + WAC_i + XAC_i$. By definition $s_i \leq q_i(i=1,---,r)$.

The file was organized with the records containing instructional data and test data all arranged sequentially in chronological order. This arrangement made it possible to compute not only the test scores but also the total cumulative time on course in minutes for each x value.

The adjustment of individual test scores (s_i/q_i) to a course wide basis was carried out in the following way. The total number of test questions posed to the student during the entire course was computed; represent this value by n. Then,

$$n = \sum_{i=1}^{r} q_i \quad \text{where} \quad q_i = CAC_i + WAC_i + XAC_i$$

Each of the values of the dependent variable $x_i (i=1, ---, r)$ is computed as,

$$x_i = (\sum_{j=1}^{i} s_i) / n.$$

The test score on the course wide basis is, for the i-th test, the ratio of the cumulated number of test questions answered correctly up through the i-th test to the total number of test questions asked of the student throughout the course.

The adjustment of the test scores in this way is necessary because the tests given within the curriculum are not cumulative. The scores are provided to the teacher and the course grade is based upon the history of test scores. It becomes the job of the teacher to cumulate the test data on individual students. In relating time on the console to performance it is absolutely necessary to have each test score defined on a comparable scale.

The purpose in using the test scores, x_1, x_2 --- x_r, as they are defined above is to gauge each student's growth of subject matter knowledge as he spends time on the console. In an extreme, if a student learned nothing beyond what he came into the course with, the test raw scores s_i would be zero (or close to it) and the time series x_1, x_2 --- x_r would all be equal valued.

REFERENCES

Chapter Four

[1] Ackoff, Russell L., and Sasieni, Maurice W. Fundamentals of Operations Research. New York: John Wiley and Sons, 1968.

[2] Brelsford, John W., Jr., and Atkinson, Richard C. "Recall of Paired-Associates As a Function of Overt and Covert Rehearsal Procedures." Technical Report No. 114, Institute for Mathematical Studies in The Social Sciences, Stanford University, Stanford, California, 1967.

[3] Hilgard, Ernest L. Theories of Learning. New York: Appleton Century Crofts, Inc., 1956.

[4] ______, and Atkinson, Richard C. Introduction to Psychology. Fourth edition, New York: Harcourt, Brace and World, Inc., 1967.

[5] Skinner, B. F. The Technology of Teaching. New York: Appleton Century Crofts, Division of Meredith Corporation, 1968.

CHAPTER FIVE

THE SCHEDULING PROBLEM FORMULATION

The relationship between a student's time on the console and his mastery of subject matter can now be used in finding a best schedule of instruction for all students in a group. One major task in producing any best schedule will be finding a definition of "best" which recognizes both the educational objectives of the teacher or educational decision-maker, and also the limits on the capacity of the computer assisted instruction system. In the formulation presented here a list of time allocations, one for each student, will be calculated by considering the effect of different amount of computer assisted instruction on each student's expected achievement.

Up to this point we have discussed only one student in connection with the instructional system. Now we will widen the discussion and speak of a group or class of n students. We assume that there are a number, c, of consoles available for use by the n students. The terms "schedule" and "time allocation" will be used synonomously and they will refer to a set of n quantities giving the fraction of each student's class time which is to be spent on one of the consoles.

As an example of how a schedule would look, imagine that we have 5 students (n=5) and that there are two consoles (c=2). In this example let the total class time consist of 50 classes, each of 45 minutes. Each student is available for instruction for 50 x 45 = 2250 minutes, at most.

Suppose we have determined on some basis, that the best schedule for the students is given in Table 5. 1 below:

Student (i)	Minutes on console (t)	% of course time on console Φ
1	500	2/9
2	1500	6/9
3	0	0
4	750	3/9
5	1000	4/9
	3750	15/9

Data: Hypothetical

Optimal Schedule: Sample

Table 5. 1

In this example the schedule is the list of Φ_i values which tells the fraction of the 2250 minutes of class time to be spent on one of the two consoles by student i.

Since there are c consoles, there is a limit to the total console time which we can allocate. This fact is expressed by the capacity constraint (1) below

$$\sum_{i=1}^{n} \Phi_i \leq c \tag{1}$$

In the example (Table 5. 1) the schedule has

$$\sum_{i=1}^{5} \Phi_i \leq 2$$

Since there are two available consoles, that schedule obeys the capacity constraint.

OBJECTIVE FUNCTION

The choice of an objective function to use in the scheduling algorithm must be made heuristically. Once an objective function is used in building the scheduling algorithm, a solution produced by the algorithm is the best one that can be found under the constraints and conditions which underlie the algorithm. The semantics of the word optimal can cause misunderstandings, however. The schedule is optimal only within the restricted context of the computerized instructional system and the group of students being scheduled. The objective function covers all students being considered for CAI although not all may be ultimately scheduled for console time.

There are some general concepts which can be used in selecting an objective function. The two main considerations are: (1) even though we use a summary measure for the whole group, the instruction is given to individuals, (2) the objective function must recognize each person's needs in achieving the instructional goals. A solution is said to be optimal if the mathematical optimality of the schedule is attained by recognizing each individual's needs to achieve the operational performance goals.

The main basis available for operationally stating educational goals is the student's mastery of subject matter which we measure by giving him a test. If, by the end of the course, a student (r) has spent a time t_j on the console, we estimate his final achievement test score $x_r(t_j)$ by the learning model of the preceding section. When there is a fixed length of class time, T, in which the course must be completed, we will write $x_r(\Phi_j)$ in place of $x_r(t_j)$. Then,

$$t_j = \Phi_j T \quad .$$

For a given schedule $\{\Phi_i, i=1,---,n\}$ we estimate the class's achievement test scores $\{x_i(\Phi_i), i=1,---,n\}$. From this vector of test scores we decide whether or not the time allocation $\{\Phi\}$ is a good one. To do this we must combine the test scores into a performance or objective function. Once we have this objective function, mathematical optimization techniques can be used to find an allocation of instructional time $\{\Phi_i^* ;\ i=1,---,n\}$ that is best in terms of the objective function and any constraints on available resources and allowable allocations.

One way of combining and summarizing achievement scores is the mean or average value. Teachers and educators are accustomed to thinking in terms of class averages (they commonly describe the performance of a class by citing the average grade). In selecting students for computerized instruction, however, it is not enough to deal only with class averages. If raising the anticipated class average were our only concern we might neglect a number of students altogether in order to devote all of our resources to those students expected to make the largest improvement in the class average, given the resources available.

There are many ways to combine the expected student achievement results into an objective. Several of the more important are:

(1) Maximize the sum of the n student's achievement scores.

(2) Maximize the average achievement score of the class minus some function of the variance of the class scores.

(3) Maximize the α percentile of the class grade distribution, where $\alpha < 0.50$ (Fractile Criterion).

(4) Maximize the probability that all n students score above some passing grade, i.e. that all students pass.

(5) Maximize the average class achievement score subject to a constraint on the probability that each student might fail (fall below a critical score).

There are certainly many others which are not listed.

We have already said that maximizing the class average alone may lead to an educational path of least resistance which would ignore those students who are not expected to progress very rapidly. Thus; (1) a performance criterion which predictably ignores any group completely in generating the schedules is not acceptable. There are two other factors which will be considered in selecting a performance function. (2) A performance function must be as close as practicable to expressing the teacher's or educator's objectives. (3) Any performance criterion which is to be used must be operationally definable and translatable into an equivalent mathematical function so that optimization techniques can be used to find improved schedules.

The first performance alternative that of maximizing the sum (or mean) of the classes' achievement scores is unsatisfactory. It does meet the condition that it should be meaningful to teachers and that it should be mathematically tractable. Its fault is that it would lead to schedules that ignore students expected to move slowly on the curriculum.

The second alternative is to maximize some function of the class mean less some function of a measure of dispersion of the class scores $\{x(\Phi)\}$. By using variance as the measure of dispersion of class scores, the problem becomes a quadratic programming exercise. This formulation could be used to generate schedules with little variation of student test scores around the class average. The advantage of this approach is that the schedules would balance off increases in class average against increased variance of expected student test scores. To do so, however, the teacher would have to be able to express his preferences for trade-offs between increased class average and increased variance of test scores. It is unlikely that this type of value information could be obtained with reliability; educators, parents and other interested parties are not accustomed to thinking in terms of variance. Furthermore, there is still a chance that students at the extremes will not be treated in a way the decision-makers would judge satisfactory.

The next alternative is to maximize a specified fractile of the class grade distribution. For example we might try to find a schedule which produces the largest value for the 10th percentile of the group's test score criterion. We might consider such a criterion if the lowest scoring ten percent of each class are failed. Then it would be equivalent to maximizing the lowest passing grade.

The fractile criterion (as this objective is called) has been studied by Geoffrion [1]. Geoffrion's work on aspiration level and fractile criterion programming problems has been motivated by problems of portfolio management in the investment area. Unfortunately, the procedure developed by Geoffrion for solving this type of problem hold only under assumptions that cannot be made here. Furthermore, the computational scheme proposed by Geoffrion for solving problems of this type is so complex that it cannot be regarded as a practical procedure in this application.

The concepts which underlie the fractile criterion have suggested the next two performance function candidates; so the desirable features of this criterion can be kept.

The fourth performance criterion is meaningful and it leads to a nice result. If we maximize the probability that all students pass (i.e. score above some preset grade on the achievement test) this is equivalent to:

$$\max_{\{\Phi\}} \prod_{i=1}^{n} p_i(\Phi_i)$$

$p_i(\Phi_i)$ is the probability that student i will pass the final achievement test if he is allotted Φ_i of the maximum time on a console.

This objective function will lead to an allocation which places increasing emphasis on students as their probability of passing goes down. For example, in a situation with 10 students where 9 are given a 0.5 probability of passing and one has 0.00 probability of passing the objective function has the value $(0.50)^9(0) = 0$. Whatever the resources available, first claim goes to the student who is certain to fail. For him, any positive improvement in this probability of passing will increase the overall objective by an infinite percentage even though he may have been the most difficult to teach.

Apparently, this alternative suffers from faults similar to those arising from the use of the mean grade as the objective. In this case the group receiving disproportionate attention are those who are likely to fail; for the mean the most progressive students are favored.

Finally we turn to the fifth alternative: to maximize the mean subject to constraints on the probability that each student might fail. This approach uses the measures class average and failure probability both of which are familiar to educators. It is mathematically tractable and it takes the development of each student into account.

The teacher must specify the test score k that is the lowest passing grade on the final achievement test and also state the largest acceptable probability that a student may fail, α. A teacher may be reluctant to state a failure probability for a student or a class of students. If that is so we might ask the teacher to specify the largest number of students, out of the class of size n, that could ever be permitted to fail. From this we could infer the probability α. For example, if an educator will tolerate no more than one failure in a class of 20 we would set $\alpha = 1/20 = 0.05$.

The console time allocation procedure which we now develop will be designed to find $\{\Phi^*\}$ which gives the largest class average test score with the constraint on failure probabilities. This formulation has an accepted technical name: chance constrained programming. An objective is being optimized subject to constraints on the chance of certain events (failures).

ALLOCATION PROCEDURE: DEVELOPMENT

Suppose that the class of n students has completed its course of instruction according to a schedule $\{\Phi_i ; i=1,---,n\}$ and suppose that we have administered the final achievement test. For student i his actual score is the fraction of the N achievement test questions which he answers correctly. Let s'_i represent this score.

By assumption 7 of Chapter Three the quantity s'_i is a random variable distributed binomially with a mean of $x_i(\Phi_i)$ and a variance of $x_i(1-x_i)/N$. [Recall that $x_i(\Phi_i)$ is the expected score on the final achievement test after completing the proportion Φ_i of the course time on the console. From here on we write x_i instead of $x_i(\Phi_i)$ where ever practical]. As long as x_i is between 0.10 and 0.90 and $N > 30$ the binomial distribution of s'_i can be approximated accurately with a normal density function. We shall use the approximation

$$f(s'_i) = \frac{\sqrt{N}}{[2\pi x_i(1-x_i]^{1/2}} \exp\left[\frac{-N}{2}\frac{(s'_i - x_i)^2}{x_i(1-x_i)}\right] \quad 0 < s_i < 1$$

or in shorthand notation

$$f(s'_i) \ : \ \text{Normal}\ (x_i, \frac{x_i(1-x_i))}{N} \qquad 0 < s_i < 1$$

The objective is to maximize the average of the s'_i taken across students subject to the constraint that the probability that any student fails is equal to a specified value α_i. For generality we will allow a separate failure probability for each student, although we may actually have one failure probability for all.

When we are choosing a time allocation (a set of Φ's) we cannot observe the actual test scores s'_i so we replace them with their expected values $x_i(\Phi_i)$. The problem is now;

$$\underset{\Phi_i}{\text{Max}} \ \frac{1}{n} \sum_{i=1}^{n} x_i(\Phi_i) \tag{2}$$

subject to the restrictions that the probability of scoring below k is α_i;

$$\text{Prob. } \{s'_i < k\} = \alpha_i \qquad i=1,2,---,n \tag{3a}$$

$$0 < s'_i < 1 \qquad i=1,2,---,n \tag{3b}$$

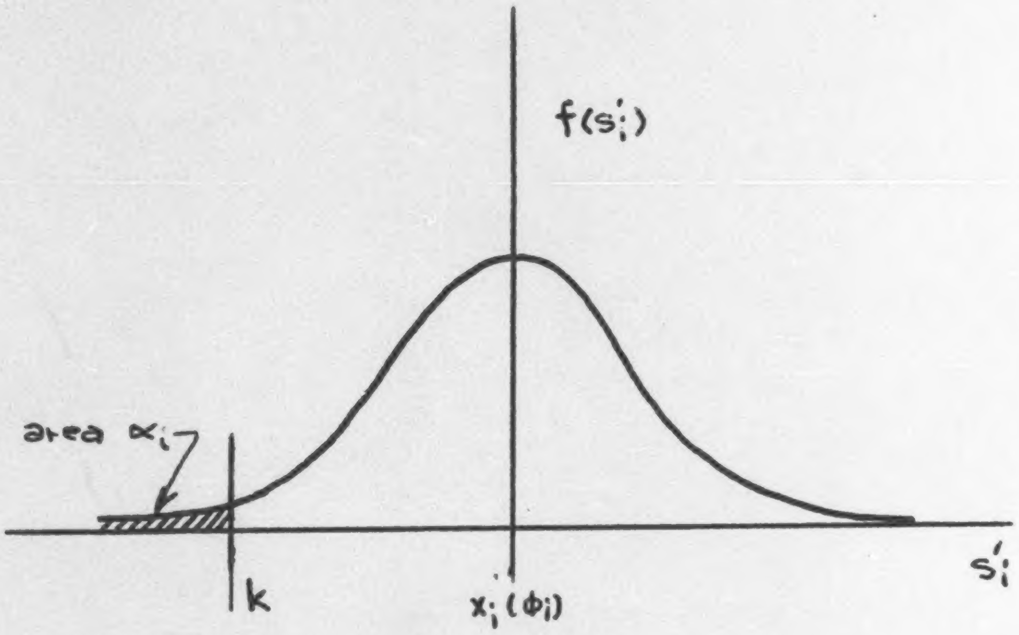
By using the normal approximation we can rewrite (3a) as:

$$\alpha_i = \int_0^k \frac{N^{1/2}}{2\pi[x_i(1-x_i)]^{1/2}} \exp\left[\frac{-N}{2}\frac{(s'_i - x_i)^2}{x_i(1-x_i)}\right] ds'_i \qquad \begin{array}{l} 0 < s'_i < 1 \\ i=1,2,---,n \end{array}$$

The integral is taken from 0 up to k, the passing grade, rather than from $-\infty$ up to k, because negative values of the test score s'_i cannot occur. We also drop the 1/n from the objective function and rewrite (2) as

$$\underset{\{\Phi\}}{\text{Max}} \sum_{i=1}^{n} x_i(\Phi_i) \tag{2a}$$

The constraints in (3a) can be re-expressed in a simpler form. The figure below illustrates the distribution of s'_i about its expected value $x_i(\Phi_i)$.



Distribution of Actual Score Around Mean

Figure 5.1

The probability that the i-th student's actual achievement test score s'_i is less than the passing grade cutoff k is equal to the probability that

$$\frac{N^{1/2}(k-x_i)}{[x_i(1-x_i)]^{1/2}} \le Z_{\alpha_i}$$

where Z_{α_i} is defined as the α_i percentile of a standard unit normal variate. For a chosen value of α_i, Z_{α_i} is a constant which may be found in a normal table.

So,

$$\text{Prob}\ \{s'_i \le k\} = \text{Prob}\left[\frac{N^{1/2}(x_i-k)}{[x_i(1-x_i)]^{1/2}} \ge Z_{\alpha_i}\right] = \alpha_i \tag{4}$$

Provided that $x_i > k$, squaring both sides of the second inequality in (4) does not change the inequality and we get,

$$(N + Z^2_{\alpha_i})\, x_i^2 - (2kN + Z^2_{\alpha_i})\, x_i + Nk^2 \ge 0 \tag{5}$$

Since this is a quadratic we solve for the larger root

$$x_i \ge \left[\frac{2kN + Z^2_{\alpha_i} + Z_{\alpha_i}\sqrt{Z^2_{\alpha_i} + 4Nk(1-k)}}{2(N + Z^2_{\alpha_i})}\right] \equiv \pi_i \tag{6}$$

This inequality is useful because the right hand side is just a constant, π_i, so the inequality is linear.

To illustrate this suppose that for a particular student we allow a probability of 0.05 ($\alpha_i = 0.05$) of failure on a test of 50 questions ($N = 50$) where the lowest passing grade is 0.70 ($k = 0.70$). The stated α_i of 0.05 has a corresponding Z_{α_i} of -1.65 and the expression (6) gives the result

$$x_i \ge 0.7962$$

So, for specified values of α_i, k and N, all of the probability constraints

(3) transform into simple deterministic constraints of the form

$$x_i \geq \pi_i \; . \qquad\qquad i=1,---,n \qquad (7)$$

Each student could have a different π_i value, but all π_i's are constants.

In the earlier development of the learning model we developed a relationship between each student's time allocation Φ_i and his expected test score x_i. The test score x_i is given by

$$x_i(\Phi_i) = \frac{e^{k'_{1,i}\,\Phi_i - k_{2,i}}}{1 + e^{k'_{1,i}\,\Phi_i - k_{2,i}}} \qquad\qquad 0 \leq \Phi_i \leq 1$$

Thus we can write the constraints (7) as a function of Φ_i directly.

$$x_i \quad = \frac{e^{k'_{1,i}\,\Phi_i + k_{2,i}}}{1 + e^{k'_{1,i}\,\Phi + k_{2,i}}} \geq \pi_i \qquad\qquad i=1,2,---,n$$

Rewriting, simplifying and taking logarithms we have

$$\Phi_i \geq \frac{1}{k'_{1,i}} \left[\ln\left(\frac{\pi_i}{1-\pi_i}\right) - k_{2,i} \right] ; \qquad\qquad k'_{1,i} \geq 0 \qquad (8)$$

$$i=1,---,n$$

Once we have evaluated each student's π_i and have obtained estimates of each student's $k'_{1,i}$ and $k_{2,i}$ the constraints in (8) place lower bounds on the amount of time which each student may be assigned. If we set small α's (very low probabilities that any student should fail) or if we set a high passing test score k, it is possible that the total instructional time required to satisfy (8) might exceed the total computer time available. In this case there would be no feasible schedule.

The Allocation Problem

The original problem of allocating computer console time to the student has now been formulated mathematically. We wish to maximize the average score on an achievement test given to all n students in a group. At the same time we want to specify the probability that a student will pass his final achievement test on the subject matter, which translates into minimum values of Φ_i. Finally, the total time allocated to the n students shall be less than or equal to the available time.

This may be expressed as

$$\left.\begin{array}{lll}
\text{Find} \quad \underset{\{\Phi\}}{\text{Max}} \sum_{i=1}^{n} x_i(\Phi_i) & & \text{(2a)} \\
\text{subject to the constraints} & & \\
\Phi_i \geq \dfrac{1}{k'_{2,i}} \left[\ln\left(\dfrac{\pi_i}{1-\pi_i} \right) - k_{2,i} \right] & i=1,---,n & \text{(8)} \\
\sum_{i=1}^{n} \Phi_i \leq c & & \text{(1)} \\
\Phi_i \geq 0 & i=1,---,n &
\end{array}\right\} \quad (9)$$

The optimal solution will be written

$$\{\Phi_i^* \; ; \; i=1,---,n \} \equiv \{\Phi_1^*, \Phi_2^*,---,\Phi_n^* \} \equiv \{\Phi^*\}$$

This chance constrained optimization problem (9) consists of maximizing a non-linear objective function subject to 2n + 1 linear restrictions. Each element in the objective function is stated in terms of one variable only, Φ_i; in mathematical terms the objective function of this problem is made up of separable functions of the various Φ_i.

A very general approach to this type of optimization problem has been developed. The technique of separable programming enables us to generate a new problem which is a piecewise linear approximation to (9) [2]. The approximate problem can then be solved by a modified linear programming algorithm.

Before any further discussion of the properties of the problem that permit a solution of (9) by separable programming, we shall look at the interpretation of the final solution to (9). The method of finding the solution is discussed and illustrated in Chapter Seven.

A solution to the time allocation problem (if one exists) will provide a set of values for $\{\Phi_i^*; \; i=1,---,n\}$. The schedule, if it is carried out, will maximize the forecasted overall class average on the final achievement test at the same time that each student has a fixed probability α_1 of achieving below the failing grade k on the test.

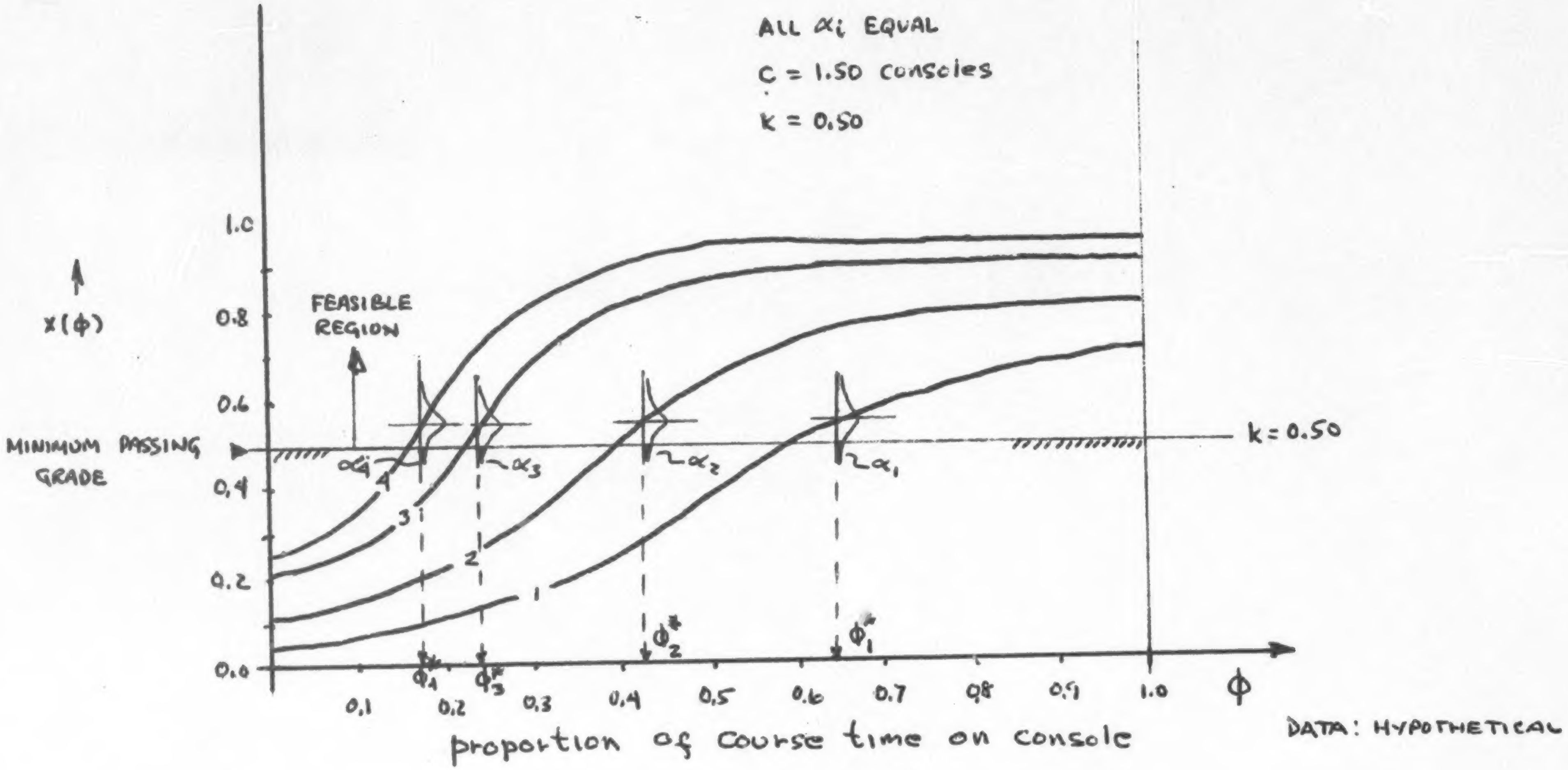
Beside $\{\Phi^*\}$ the solution to (9) will give us valuable information on how the objective function (2a) is effected by changes in the α_i values or in the number of consoles c. By looking at the shadow prices on each of the constraints (8) and on the capacity constraint (1) we can estimate the effect on overall class performance of adding more console time.

Illustration of Optimal Solution

To illustrate the consequences of the scheduling algorithm suppose that we have a group of four students and that the parameters k_1' and k_2 are known for each of them. To make the example specific let us set the passing grade at 0.50 (k = 0.50), and let the allowable probability of failure (probability of a grade of k or less) be the same for each student.

$$\text{Prob}\,[x_i < k] = \alpha_i = 0.10 \qquad i=1,---,4$$

We shall also assume that initially 1.5 consoles are available for these students' use. [This might happen if one of the consoles is tied up for half of the time by another group]. As can be seen in Figure 5.2 the problem has a feasible solution which allows each student to satisfy the failure probability constraint, but it does not allow for any of the students to move much above the minimum feasible. If a student (say #4) were to have more time, one of the remaining three would have his allocation reduced because the amount of console time is



DATA: HYPOTHETICAL

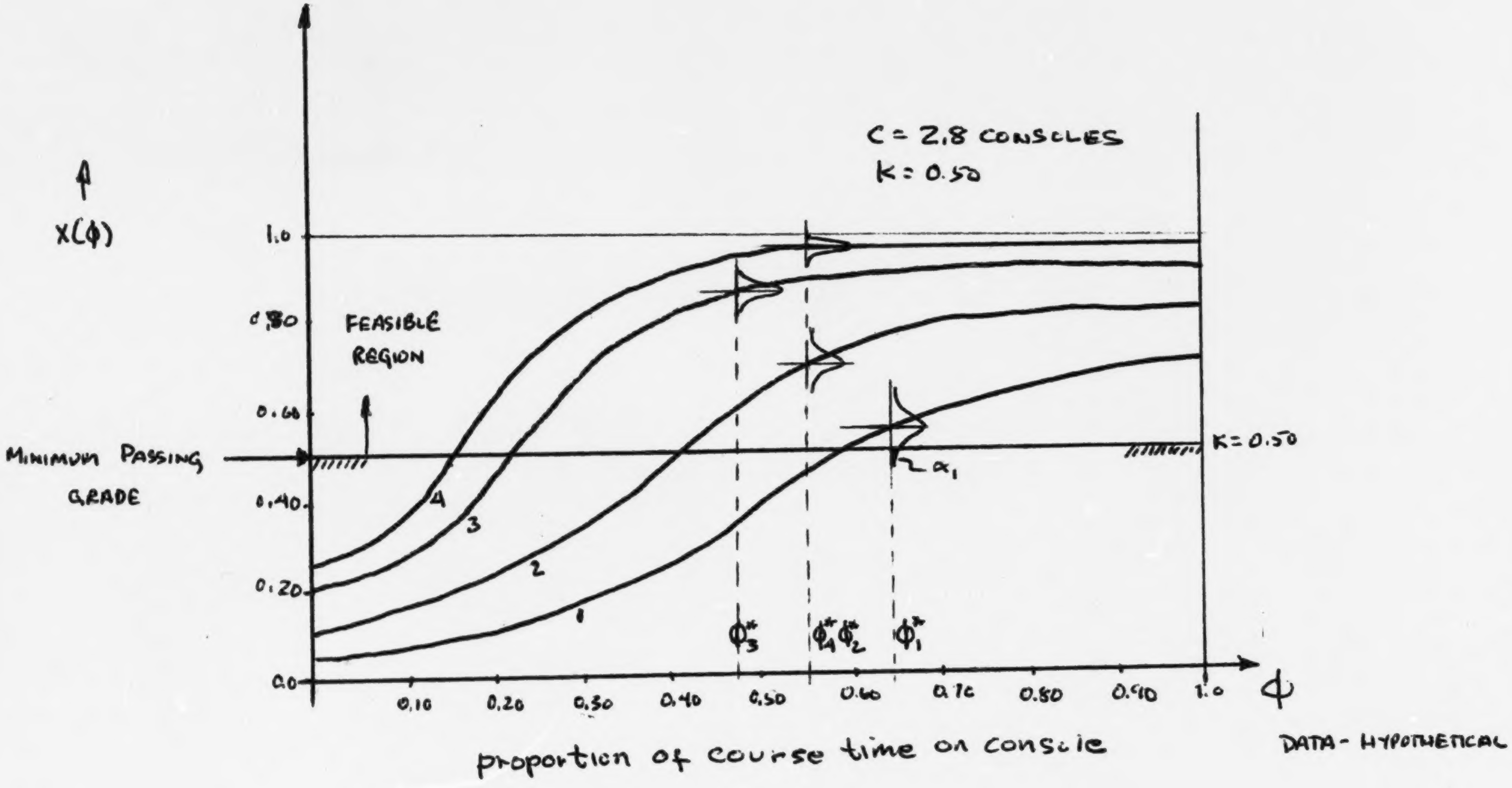
The Effect of Chance of Failure Constraints on Feasible Schedules

Figure 5.2

fixed. Although the average grade of the four might be increased by giving more time to 4, one of the others would have a chance of achieving below 0.50 on the achievement test greater than 0.10.

In this example the shadow prices associated with the console capacity would indicate that very substantial improvements in class average can be obtained by acquiring more console time. The shadow prices would also indicate that a very large improvement in class average would result if we allowed student number 1 to have an increased probability of failure. This would make an amount of console time available for the others, but it would not reduce x_1 by as much as the values of x_2, x_3, and x_4 would go up on account of the added console time allocated to them. Examples of shadow prices will be given in Chapter Seven.

Now suppose that with other things being the same we obtain 1.3 added consoles. Now 2.80 consoles are available for the class of four. The effect of this change is illustrated in Figure 5.3. Student #1 does not have any change in his schedule time, however the others have marked increases in their expected scores as a result of having obtained larger time allocations. Now there would be a much smaller added increase in overall average achievement test score if we could obtain even more console time.



The Effect of Additional Console Capacity on Feasible Schedules

Figure 5.3

The effect of this allocation procedure is to start with a student time schedule (based on the estimated parameters of each student's characteristic learning curve) which is expected to satisfy the constraints on individual failure probability. After a feasible schedule is achieved any remaining console time resources are used to drive the class average test score up at the highest possible rate.

We know that the class average will be as large as possible under $\{\Phi^*\}$ but we can also draw some useful conclusions about the approximate number of students who fail (score below k). Since the probability that each will fail is less than or equal to α_i the expected number of failures under the optimal policy will be less than or equal to n_f where

$$n_f = \sum_{i=1}^{n} \alpha_i$$

In the special case where we give each student a constant chance of failure (α) we find an interesting result. In that case n_f is binomially distributed with distribution

$$f(n_f) = \frac{n!}{n_f!\,(n-n_f)!} \quad \alpha^{n_f}(1-\alpha)^{n-n_f} \qquad n_f = 0, 1, ---, n$$

In this special case the upper bound on the expected number of failures is $n \cdot \alpha$ and the variance is given by $n \cdot \alpha(1-\alpha)$. Of course the probability that all students pass with an achievement test score of k or greater is at least $(1-\alpha)^n$. When the α_i are different for different students the distribution of n_f is multinomial.

This distribution of the number of failure suggests a test which could be run on the whole procedure. Once a schedule is computed and implemented the actual number of failures may be compared with the number of expected failures, (calculated from the α's). A significance test could then be used to test the hypothesis that the allocation procedure's failure constraints have been satisfied.

PROPERTIES OF THE SOLUTION TO THE ALLOCATION PROBLEM

The separable programming approach to solving the allocation problem (7) uses an approximation to the original problem. Hadley [2] indicates that there are several questions to be looked into when separable programming is applied. They are: (1) the mathematical properties of the set of feasible solutions and of the objective function must be checked to find out whether or not the solution found to the approximate problem is also the global optimum solution in the original non-linear problem; (2) the accuracy of the solution in the approximating problem depends upon how well the original functions are approximated by the linearized functions.

The second point is easy to deal with in our problem. All of the functions which require approximation (the $x_i(\Phi_i)$ functions in the objective function) are very smooth. They can be approximated to very high accuracy with very few line segments. A procedure for generating these approximations is discussed in Chapter Seven.

Now we look at the mathematical considerations. Under certain conditions the optimal solution to the linearized approximation may not be the overall best solution to the allocation problem (9). An optimum in the approximate problem is the best solution to the exact problem (9) only if the exact problem's objective function is concave and the set of all feasible solutions is convex [2].

The objective function is a sum of non-linear functions. The objective will be concave if each of the individual functions is concave. A necessary and sufficient condition for this is that

$$\frac{d^2 x_i(\Phi_i)}{d\Phi_i^2} < 0 \quad ; \qquad i=1,2,---,n.$$

We already have from the original derivation of $x_i(t_i)$ or $x_i(\Phi_i)$ that since Φ is linearly proportional to the original t

$$\frac{dx}{d\Phi} \sim x(1-x)$$

Thus

$$\frac{d^2x}{d\Phi^2} \sim \frac{dx}{d\Phi}(1-x) - x\cdot\frac{dx}{d\Phi}$$

$$\sim x(1-x)^2 - x^2(1-x)$$

and

$$\frac{d^2x}{d\Phi^2} \sim x(1-x)(1-2x)$$

so $d^2x/d\Phi^2$ is negative if and only if $1/2 < x < 1$. This means that if each student's expected achievement test score is greater than 1/2, i.e.

$$x_i > 1/2 \qquad i=1,---,n$$

then the objective function will be concave. Because of the constraints on the probability that the student shall pass, if the passing grade is greater than or equal to 1/2, $(k \geq 1/2)$ the x_i values will always be larger than 1/2. So, if a feasible solution $\{\Phi\}$ exists for the passing grade value of $k = 0.50$ or more, the objective function will be concave.

Because the constraints in (9) are all linear inequalities, and since an upper and lower bound is specified for each $\Phi_i(i=1,---,n)$ the set of feasible solutions is a bounded convex set.

Since the convexity and concavity conditions on the constraint set and on the objective are satisfied and since the non-linear functions can be accurately approximated, the solution to the approximate problem will be the global optimum for problem (9). If it were necessary to allow x_i to take values lower than 1/2 the global optimality of the separable programming solution may be in doubt. In such a case there are features built into the separable programming algorithm which enable checks to be made on the solution.

The possibility of an exact method of solution for (9) was checked by deriving the Kuhn-Tucker necessary conditions (which are also sufficient in this case). The solution of these conditions, if it could be found, would provide the exact solution to (9) without resorting to approximate methods. (This derivation is included as Appendix 5. 1).

The Kuhn-Tucker conditions do not say much in this case, and they appear to be more difficult to work with than the original problem. This reinforces the use of separable programming as a practical method of solution to the allocation problem.

APPENDIX 5. 1

CHAPTER FIVE

Derivation of the Kuhn-Tucker conditions for the allocation problem (9).

The problem rewritten here is:

$$\left.\begin{array}{ll} \text{Find} \qquad \max_{\{\Phi\}} \sum_{i=1}^{n} [1 + e^{c_i\Phi_i + c'_i}]^{-1} & \\ \text{subject to the constraints,} & \\ -\Phi_i \le -f_i \; (f_i \text{ constants}) \; ; & i=1,---,n \\ \sum_{i=1}^{n} \Phi_i \le c & \\ \Phi_i \ge 0 \;\; ; & i=1,---,n \end{array}\right\} \quad (9)$$

We now derive the Kuhn-Tucker necessary conditions for the optimal solution $\{\Phi^*\}$ to (9) [2]. These conditions are also sufficient due to concavity of

$$\sum_{i=1}^{n} x_i(\Phi_i).$$

Form the Lagrangian function, $F(\Phi, \lambda, \mu)$; Φ_i and λ_i are typical elements of the vectors Φ and λ.

$$F(\Phi, \lambda, \mu) = \sum_{i=1}^{n} [1 + e^{c_i\Phi_i + c'_i}]^{-1} + \sum_{i=1}^{n} \lambda_i(\Phi_i - f_i) + \mu(c - \sum_i \Phi_i)$$

Let $\Phi^* = \{\Phi_i^* ;\ i=1,2,---,n\}$ be the point where

$$\sum_{i=1}^{n} x_i(\Phi_i^*) = \underset{\{\Phi\}}{\mathrm{Max}} \sum_{i=1}^{n} x_i(\Phi_i)$$

Then (following Hadley [2]), we require that for Φ^*

$$\frac{\partial}{\partial\Phi_i} F(\Phi, \lambda, \mu) \Big|_{\Phi^* \lambda^* \mu^*} \leq 0 \qquad i=1,---,n$$

$$[\frac{\partial}{\partial\Phi_i} F(\Phi, \lambda, \mu)] \cdot \Phi_i \Big|_{\Phi^* \lambda^* \mu^*} = 0 \quad ; \quad i=1,---,n$$

We also require that for λ_i^*

$$[\frac{\partial}{\partial\lambda_i} F(\Phi, \lambda, \mu)] \ \lambda_i \Big|_{\Phi^*, \lambda^*, \mu} = 0 \quad ; \quad i=1,---,n$$

$$[\frac{\partial}{\partial\mu} F(\Phi, \lambda, \mu)] \ \mu \Big|_{\Phi^*, \lambda^*, \mu} = 0$$

We must also include the non-negativity conditions below

$$\left.\begin{array}{ll} \Phi_i - f_i \geq 0 & ; \ i=1,---,n \\ \text{and} & \\ c - \sum_{i=1}^{n} \Phi_i \geq 0 & \end{array}\right\} : \text{Non-negativity conditions}$$

These conditions are necessary for an optimum and by the concavity of the objective function (over the constraints space) they are also sufficient.

In summary, the Kuhn-Tucker conditions for an optimal solution to problem (9) are:

$$\left.\begin{array}{ll}
\dfrac{-(c_i e^{c_i\Phi_i^* + c_i'}}{\left[1 + e^{c_i\Phi_i^* + c_i'}\right]^2} + \lambda_i^* - \mu_i^* \leq 0 & \\[2ex]
\{\Phi_i^* [1 + e_i^{c_i\Phi_i^* + c_i'}]^{-2} \, (c_i e^{c_i\Phi_i^* + c_i'}) - \lambda_i^* \Phi_i^* + \mu^* \Phi_i^* \} = 0 & \\[2ex]
\text{for } i=1,---,n, \text{ and:} & \\[2ex]
\lambda_i^* (\Phi_i^* - f_i) = 0 & ; \quad i=1,---,n \\[2ex]
\mu^* (c - \sum_{i=1}^{n} \Phi_i^*) = 0 & \\[2ex]
\Phi_i^* - f_i \geq 0 & ; \quad i=1,---,n \\[2ex]
c - \sum_{i=1}^{n} \Phi_i^* \geq 0 &
\end{array}\right\} (10)$$

Unfortunately these conditions do not suggest any straightforward means for finding $\{\Phi^*\}$. This suggests that the separable programming algorithm may be the most direct way to find the solution $\{\Phi^*\}$. Other approximate methods for finding $\{\Phi^*\}$ might be derived from (10).

REFERENCES

Chapter Five

[1] Geoffrion, Arthur M. "Stochastic Programming with Aspiration or Fractile Criteria." Management Science, Volume 13, No. 9, May 1967, pp. 672-679.

[2] Hadley, George. Non-Linear and Dynamic Programming. Reading, Massachusetts: Addison Wesley Publishing Co., 1964.

CHAPTER SIX

ESTIMATION OF STUDENT LEARNING PARAMETERS

INTRODUCTION

The validity of any schedule of computerized instruction depends upon the accuracy of the information upon which the schedule is based. The procedure of the last chapter relies upon estimates of the two learning model parameters (k_1' and k_2) for each student. These values summarize the available information that is relevant to the scheduling decision process.

Some relevant data is better than no data at all in deciding on console scheduling assignments. Smallwood [23] has studied the question of how much data should be obtained in making instructional decisions, though this work has not yet produced applicable results. As a strictly practical matter, however, the question of how much data to gather (or how much effort to expend on estimating k_1' and k_2) must be answered on the basis of experience, and on the basis of resource constraints and administrative conditions in the school environment.

The value of data gathering activity may be assessed from one of two points of view. Where data is gathered and manipulated to answer a research question, as in testing the learning model, a highly specialized effort is justified. On the other hand, specialized or complicated data collection and manipulation is out of place for making routine decisions in the school. We can be sure that the scheduling procedure will be used improperly or not at all if it requires data not readily obtainable from existing school sources.

AN IDEALIZED DATA GATHERING PROCEDURE

There is a way to use the computer system as the main data gathering tool, though this method is an ideal, rather than a practical alternative.

Imagine that every student being considered for CAI is required to spend some time (for example three hours) on a special curriculum prior to being given his ultimate console time schedule. We would then gather three hours worth of machine readable data on each student's interaction with the computerized curriculum. This data could be processed and the relevant parameter estimates could be punched onto cards for input to the scheduling routines.

The main costs of this procedure are the student time expended as well as the computer and console hours consumed in gathering the data. By using the consoles to gather data strictly for decision-making we lessen the amount of console time that we can ultimately allocate to students who need it. The improvement of scheduling decisions gained by using console time for data gathering must be weighed against the adverse effects of reducing instructional console time available for all.

Since at some schools there are several times as many students as can be accommodated on available consoles the ideal procedure outlined above is not practical. Interviews with Project GROW personnel reveal that it is politically unacceptable to put all students onto consoles and then later to reassign some to other classes. The emphasis in developing parameter estimation procedures has therefore been on developing estimators of scheduling algorithm parameters that are functions of data available outside of the computerized system.

Types of Variables

To estimate the parameters used in the scheduling algorithm one would like to use variables which explain the parameter values in the models. This is the approach strongly recommended by Ackoff [1]. But we are far short of understanding the causes of the learning process and so it is necessary to look beyond the few obviously relevant variables. There are many variables which qualified observers believe to be partial determinants of learning behavior.

These two classes of variables will be referred to as (a) causal and (b) indirect variables whenever it is necessary to make a distinction. In CAI, reading ability would be a causal variable. If a student cannot read at some minimum standard he cannot interact effectively with the computer administered instruction. Past student performance would be an indirect variable since it is presumably the result of underlying (and unknown) causal variables.

Identifying Variables

The problem is first to identify variables which are either causally or indirectly relevant and then to study them in order to learn about their relationship to the learning behavior of interest. Although much has been written about student performance on computerized instruction there is still very little really known other than; (a) that students can be placed before CAI consoles, (b) that they will learn varying amounts at varying rates. In a thorough review of the literature, Gentile [7] could report very little beyond the obvious. Stolurow [24] in his early book gave some indications (e.g., motivation and memory span.)

After a thorough survey of the literature* and discussions with other investigators studying problems related to the estimation problems of this thesis, the study was narrowed down a few variables.

---

* [2, 3, 4, 5, 6, 7, 9, 10, 11, 12, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28].

One of the distinguishing characteristics of CAI is that the pace of learning is controlled by the student. It is well known that students vary greatly in the rates at which they complete self paced activities [29].

Two approaches were taken to estimating k_1' and k_2 :

- a special "selection aid" or pretest was designed. It is a short paper and pencil exercise designed to be easily administered before assignments to CAI are made
- data from the students' files normally maintained by the school were analyzed for their value as estimators.

These two approaches are discussed in the next two sections.

AN EXPERIMENTAL DATA GATHERING PROCEDURE

A "selection aid" consisting of a piece of programmed instruction and an achievement test was developed in cooperation with the Summer Reading Program at Wanamaker Junior High School. This experiment is discussed in detail in this section.

An overwhelming body of experience points to the rate of student progress over curriculum as being a crucial student variable. This variable related directly to k_1' in Model II of Chapter Four. The analysis of k_1' in Chapter Four shows that this time coefficient in the learning model varies widely from student to student. This is a very important fact since the constant k_1', estimated for each student, enters directly into the constraints and objective of the mathematical programming algorithm in Chapter Five (see, for example, equations 5.8 and 5.2a).

The first effort to find an off-line estimator of this rate variable was carried out during the Summer of 1968 at the Wanamaker Junior High School, Summer Reading Program. A sample of programmed text from Glassman [8] was given to all 33 student participants in the reading improvement program. (Four classes of approximately 8 each.) The programmed instruction using a paper and pencil instrument was a very close analogy to the computerized interaction. A short quiz on the material covered by the programmed instruction was also administered in the experiment.

The students wrote out their responses to the frames of programmed instruction. As they checked their responses in the feedback portion of the frame, if the response was wrong, the students crossed out the wrong answer, reread the frame, and responded again. The instructions were designed so that the second reading of the teaching point (after a wrong answer) would approximate a branch to remediation on the computerized curriculum.

In order to insure that all of the students understood how to do the programmed instruction, the teacher talked each class through a sequence of eight frames that illustrate and explain the directions. The students were then started on the sequence of thirty-five frames and allowed to work for 15 minutes. Any student finishing in less than 15 minutes was noted and very careful records of the times involved were made by observing the class through a one way mirror. At the end of the fifteen minute period of programmed instruction the booklets were collected and the quiz was administered.

From the data gathered with the selection aid (programmed instruction plus the quiz) values of several variables were calculated for each student. Values were calculated for: (a) average time per frame of instruction (seconds/frame), (b) percentage of frame responses correct first time, (c) proportion of quiz questions correct.

As a step toward examining the use of file data, Iowa mathematics and verbal achievement test scores (in grade equivalents) were obtained for each student from the School District's Division of Research. These tests had been given to the students during the preceding Spring.

The students then received six weeks of CAI training in reading techniques. At the end of the course the data gathered from the computer instruction was obtained from School District Instructional Systems personnel. This will be designated as "on-line" data.

Although every student in the reading course (33 in all) had taken the selection aid, only fourteen students had actually received enough computer assisted instruction so that significant data was available to them. This situation resulted mainly from Summer School staffing problems at Wanamaker Junior High School and the fact that student attendance at summer school (not just in the reading program) was voluntary. Thus the number of data points for analysis was reduced. An attempt was made to correlate the pre-CAI data (off-line) with CAI results.

The sample correlation matrix of the data is presented below in Table 6. 1. At some increased risk of error the estimated pair-wise correlation coefficients have been tested for significance with univariate tests. The most important section of the matrix is the lower left hand quadrant; this section contains the estimated correlations of the off-line variables (both selection aid and Iowa scores) with the on-line data taken from tapes created during the computerized instruction.

The only significant on-line/off-line correlation is between on-line response rate (latency) and the Iowa reading achievement test

	ON LINE		OFF LINE			
	From Topic Summary Tape		From Selection Aid		Iowa Tests	
	(1) % RESP. COR.	(2) RATE OF RESP. (Sec./ frame)	(3) % RESP. COR.	(4) RATE OF RESP. (Sec./ frame)	(5) READING	(6) ARITH.
(1)	1.00					
(2)	0.36	1.00				
(3)	0.16	0.35	1.00			
(4)	-0.35	0.41	-0.37	1.00		
(5)	0.08	-0.54*	-0.31	-0.28	1.00	
(6)	0.15	-0.24	-0.52*	-0.38	0.35	1.00

Note: * indicates Reject H_0: $\rho = 0$ at 5% level

Sample (n = 14) Correlation Matrix From Experimental Selection Aid And Wanamaker Summer Reading Program

Table 6.1

DATA DEFINITIONS

(1) % responses correct during CAI [CAC/CAC + WAC + XAC] whole course
(2) Average response time (whole course) weighted average of ARL counter
(3) % of selection aid programmed instruction responses correct
(4) Average response time on programmed instruction [# of seconds/ # of frames]
(5) Iowa Reading achievement score in grade equivalents
(6) Iowa Arithmetic achievement score in grade equivalents.

score ($r_{5,2} = -0.54$). Little confidence can be placed in this estimated correlation coefficient. The 95% confidence interval estimate of $\rho_{5,2}$ is approximately (-0.06 to -0.82). With the small sample size ($n = 14$) a correlation coefficient of -0.06 is as consistent with the results obtained as a correlation coefficient of -0.82.

The most surprising result was that there was no significant correlation ($r_{4,2}$) between rate of responding on the short piece of programmed instruction and rate of responding during on-line instruction. A strong correlation was expected. There are many a posteriori rationalizations for this failure of the expected correlation to appear. It is virtually certain, though, that no conclusions should be drawn until a larger sample of programmed instruction can be given to a much larger sample of students.

The inverse correlation between selection aid proportion of correct responses and arithmetic achievement ($r_{6,3} = -0.52$) was not expected. It has no immediate relation to the estimation of on-line learning characteristics with off-line data.

OFF-LINE PREDICTION OF LEARNING MODEL PARAMETERS FIRST ATTEMPT

In Chapter Four the learning-model was tested on data gathered from the on-line instruction of twenty Biology students at Germantown

High School. Matching off-line data was gathered from the school files on 18 of these students, the other two students had incomplete records. The data gathered on each student were: (1) School and College Achievement Test--1967 data, verbal and mathematics percentile scores, (2) attendance as a percentage of the 185 days in the school year ending June, 1968.

The hypotheses which suggested these data are as follows:

(1) Attendance should be a causal factor in determining the slope parameter k_1'. This hypothesis is based upon the observation that very poor attendance may be due to poor motivation of the student or actual avoidance behavior. This type of measure has been discussed by Webb and others [29] in their book UNOBTRUSIVE MEASURES. Other interpretations of absence behavior are suggested by Skinner [21]. Attendance may be causally related to learning rate by the fact that frequent absences impede instruction.

(2) Achievement data were included because past performance is one of the more stable predictors of future performance (in spite of the fact that it is an indirect variable.) Verbal achievement is included because it is an index of the types of skills that a student must use to learn with the present CAI system.

The sample correlation matrix was estimated for the 18 data points with five variables each (the estimated parameters k_1' and k_2

from Chapter Four together with the three off-line variables). No significant correlations were found between the parameter estimates k_1' and k_2 and the off-line data.

OFF-LINE PREDICTION OF MODEL II PARAMETERS SECOND ATTEMPT

The inconclusive results of the above efforts to find off-line predictors of the learning model parameters k_1' and k_2 made further empirical work necessary. Previous experiences show that missing data (because of high student mobility and very high in-out migration as well as some clerical error) are the real obstacles to be overcome in assembling a satisfactory data base.

Several additional hypotheses resulted from the previous inconclusive efforts. After discussing the results of the first Germantown and Wanamaker studies with Dr. Kenneth Wodtke (Penn-State CAI Laboratory), Dr. Wodtke suggested that the slope parameter k_1' be estimated by differences in past achievement test scores over time, rather than by absolute achievement test scores. The hypothesis is that students who have progressed rapidly in the past (large year to year achievement gain) would also behave similarly on the CAI consoles. Other hypotheses were formulated, especially concerning student impulsiveness and behavior at the consoles [13]. Nothing could be done within the constraints of time and resources to check these latter hypotheses on reflection-impulsivity.

The second effort to find predictors of k_1' and k_2 started with all Germantown biology students receiving CAI during the Fall of 1968. There were 64 students in all. Most of these students had been selected randomly from the list of all eligible Germantown students; only substitutions for drop-outs or disciplinary cases were not randomly selected (at most a third of the group). It was not possible to obtain pairs of achievement test scores on each student thus the major new hypothesis--that rate of change of past achievement would correlate with the observed slope parameter k_1' -- could not be tested.

The students were almost evenly split between two teachers (30 were with one teacher and 34 with the other). The following data on each student were recorded (if they were available):

Differential Aptitude Test--Spring, 1968

Raw scores and percentile scores* on

- verbal
- numerical
- spelling
- grammar

* Two normalizations, one for males and another for females.

School and College Achievement Test

Raw scores and percentile scores on

- verbal
- mathematics

Attendance for years ending 6/66, 6/67, and 6/68

. excused absences

. unexcused absences

After examining student records for all 64 students only 29 complete sets of data (list above) were found. There were varying amounts of data on 27 others and no data at all on 8 students. The students on whom no data at all was obtained were transfers from school districts in Florida, Maryland and South Carolina.

In an effort to use as much of the data as possible, observations having any four or fewer variables missing (out of twelve) were treated in the following way. If the datum was attendance for a given year, the mean of that student's attendance data for other years was substituted. For the various test scores a series of scatter diagrams of available data were plotted and these were used to estimate a student's missing variables given his available records. A new variable was added to the data set in order to control the effects of replacing the missing observations. For each student this variable was set equal to the number of estimated values in that student's vector of observed variables. This new variable was equal to zero for 29 students and had non-zero values for an additional 19 students. Thus, the number of observations was brought from 29 back up to 48.

With this much data it was possible to carry out much more reliable analyses.

The missing data replacements were carried out in the most conservative fashion. By testing the significance of the new missing-data variable a cross check was made for reliability of results. As a strict matter of judgement the loss in reliability of results from using a small number of estimated data points is completely overshadowed by the tremendous loss of information if these 19 nearly complete cases were omitted from analysis.

During the Fall, 1968 semester the Philadelphia School Districts' Division of Research developed and administered a test to all students in the biology CAI program. The Research test was a paper and pencil quiz of 40 items selected randomly from seven topic tests covering the first third of the course. Each biology CAI student was given the test when he had finished the first third of the CAI curriculum. Raw scores on this test were obtained for 37 out of the 48 students for whom off-line data is available.

The topic summary tape covering the first five weeks of biology instruction for these students was obtained from School District personnel. There was insufficient data on this tape to compute valid estimates of k_1' and k_2 for the Germantown biology CAI students. This presented a setback to efforts to develop off-line predictors of the parameters k_1' and k_2. The off-line data, together with the School District Research test, does provide considerable information on hypotheses stated earlier. Table 6.2 is the sample ($m = 37$) correlation matrix for the data. Variable #14 is the School District Research test, #15 is the missing data variable (its value equals the number of data elements in each student's observation vector).

Several important observations can be made from the correlation matrix. We notice that there is only one estimated ρ involving the missing data variable that is large enough to be considered significant. The apparent correlation between the Differential Aptitude test spelling score and the missing data variable suggests that we should be cautious about inferences involving the spelling variable and its correlates (numerical aptitude--DAT).

The row labeled 14-TEST is the most important for our purposes. The only variables that correlate significantly with biology achievement are sex and verbal ability. (Males and Females were coded 1.0 and 0.0 respectively). Thus the girls did significantly better on the biology test than the boys. A glance at the fifth row (SEX) shows that the girls are very significantly better in grammar as measured by the D.A.T. score. Thus, it is likely that the significant correlation

		DIFF. APTITUDE					S. C. A. T.	
		VERB 1	NUM 2	SPEL 3	GRM 4	SEX 5	VERB 6	MATH 7
1	VERB DAT	1.0						
2	NUM "	.351	1.0					
3	SPELL "	.199	.537**	1.0				
4	GRAM "	.245	.080	.228	1.0			
5	SEX	-.101	.021	-.251	-.543**	1.0		
6	VERB-SCAT	.164	.070	.144	.607**	-.212	1.0	
7	MATH "	.115	.515**	.237	.078	-.053	.122	1.0
8	66X	.079	.212	.054	.173	.053	.228	.056
9	66U	-.063	-.059	.106	.027	.002	-.056	-.009
10	67X	.095	-.086	.240	.008	-.100	-.027	-.203
11	67U	-.002	.063	-.035	.125	.210	.097	.061
12	68X	.276	-.015	.283	.070	.040	.218	-.247
13	68U	-.092	.152	-.040	-.138	.252	.132	.097
14	TEST	.244	.108	.189	.466**	-.487**	.446**	.180
15	M. D.	.025	.240	.394*	.155	.056	.171	.269
16	TEACH	.370	.099	.051	-.172	.082	.030	.010

** Reject Univariate Test at $\alpha = 0.01$
* Reject Univariate Test at $\alpha = 0.05$

Estimated Correlation Matrix
Off-line Data Germantown Biology Fall 1968
N = 37

Table 6.2

Table 6.2--Continued

		ATTENDANCE								
		66X 8	66U 9	67X 10	67U 11	68X 12	68U 13	CAI TEST 14	MISS DATA 15	TEACH 16
8	66X	1.0								
9	66U	.472**	1.0							
10	67X	.416	.460**	1.0						
11	67U	-.568**	.575**	.269	1.0					
12	68X	.137	.168	.588**	.139	1.0				
13	68U	.129	.298	.074	.338	.073	1.0			
14	TEST	.067	-.120	.145	.014	.258	.072	1.0		
15	M.D.	.012	.332	.161	.219	.194	-.017	.152	1.0	
16	TEACH	-.042	-.143	.166	-.277	.243	.283	.136	-.125	1.0

** Reject Univariate Test at $\alpha = 0.01$

between biology achievement test score and sex is due to an underlying difference in verbal ability rather than being caused by any differential effectiveness of the curriculum across the sexes.

The correlation coefficient between #16, the teacher variable, and the biology achievement test score is too small to be significant. In the biology curriculum, at least for Germantown High School, Fall 1968--the student's achievement test performance was not dependent upon which of two teachers they had.

The hypothesis concerning attendance as a causal variable in determining the slope parameter k_1' cannot be tested directly with the data at hand. Still, the lack of correlation between the achievement test score and attendance suggests that it is not an important factor for the 37 students on whom we have data. The attendance variable might still be regarded as important since poor student attendance is one reason why a student might not have taken the Research test.

A factor analysis was performed on the R matrix in Table 6.2. The analysis was limited to the three and four factor models only. A test of the significance of the three factor model led to its rejection as an adequate explanation of the covariance structure underlying the observation in hand. The first three factors had respectively high loadings on (1) verbal ability, (2) attendance, and (3) spelling - arithmetic, numerical ability. Evidently some other processes are influencing the observations available.

One final comment on the use of univariate tests in checking individual elements of a multivariate regression matrix. There is a greatly increased risk of committing type I errors by sequentially applying univariate tests on correlated statistics. For this reason we have not spoken of the level of significance while discussing Table 6.2. The asterisks used in that table are only to call attention to relatively large estimated correlations.

In summarizing the previous efforts to predict student learning characteristics either from a selection aid or from readily available off-line data we must say that none have been really successful. Until further research uncovers reliable predictors of student-console interaction there is little practical choice but to use a procedure that is similar to the idealized data gathering procedure. In the next section we analyze the properties of an on-line data gathering procedure.

MODEL II PARAMETER ESTIMATION PROCEDURE

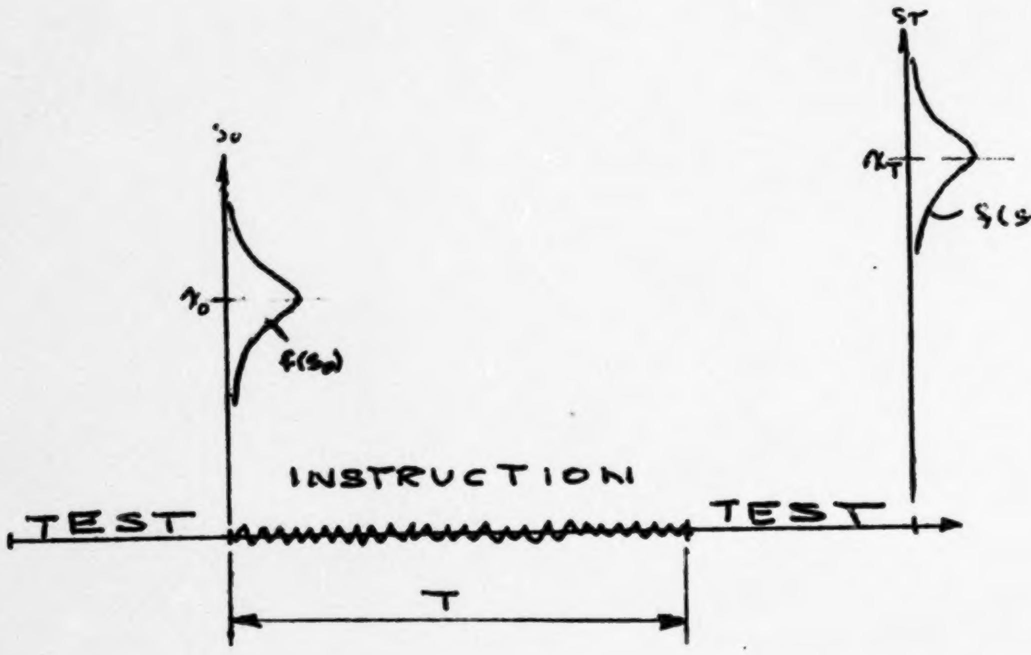
This section develops a procedure for estimating the values of k_1' and k_2. This procedure does not use off-line data but it administers a pretest to each student. The pretest would be administered to candidates for CAI by consoles at the start of a course, before any decisions had been made on scheduling. The pretest will be referred to as an "on-line selection aid."

The on-line selection aid consists of two tests separated in time by a period of instruction. Each test has N items selected so that the two scores can be adjusted to equivalent scores on the final course achievement test. The tests are over the material which is covered during the period of instruction. This sample of computerized instruction is of length T. It should be typical of the type of instruction used throughout the rest of the CAI course.

Suppose, for example, that we use the first topic of CAI instruction in this selection aid and that this topic represented P percent of the material of the course. If the raw scores of the tests are given by c_0 and c_T then we set the adjusted test scores equal to

$$s_0 = \frac{c_0}{N}\frac{P}{100} \quad \text{and} \quad s_T = \frac{c_T}{N}\frac{P}{100}$$

Figure 6. 1 below illustrates the design of this on-line selection aid. We shall represent the actual test scores as s_0 and s_T. These scores are adjusted to represent total course mastered. We now adopt the point of view that Model II of Chapter Four applies.



Schematic of the On-line Selection Aid

Figure 6.1

In this case the test scores will be regarded as realizations of random variables such that

$$E(s_0) = x_0 \qquad E(s_T) = x_T \qquad \text{and}$$

$$\mathrm{Var}(s_0) = x_0(1-x_0)/N \qquad \mathrm{Var}(s_T) = x_T(1-x_T)/N$$

From Model II of Chapter Four we have that

$$x(t) = [1 + e^{-k_1't - k_2}]^{-1}$$

so we can estimate k_1' and k_2 by $\hat{k}_1'$ and $\hat{k}_2$ with

$$s_0 = (1 + e^{-\hat{k}_2})^{-1} \quad ; \quad \text{i.e. } t = 0 \tag{1}$$

and with

$$s_T = (1 + e^{-\hat{k}_1'T - \hat{k}_2})^{-1} \quad ; \quad \text{i.e. } t = T \tag{2}$$

From (1) we have that

$$\hat{k}_2 = -\ln\left(\frac{1}{s_0} - 1\right) \tag{3}$$

From (2) after substituting (3) for $\hat{k}_2$ we find that

$$s_T = \frac{1}{1 + \left(\frac{1}{s_0} - 1\right) e^{-\hat{k}_1' T}} .$$

Now we would like to solve for $\hat{k}_1'$.

After a little manipulation we have that

$$\hat{k}_1' = \left[\ln\left(\frac{1}{s_0} - 1\right) - \ln\left(\frac{1}{s_T} - 1\right)\right]\frac{1}{T}$$

$$\hat{k}_1' = \left[\ln\left(\frac{s_T - s_0 s_T}{s_0 - s_0 s_T}\right)\right]\frac{1}{T} . \tag{4}$$

Since s_0 is typically not a very large number (0.001 to 0.05) and if T (the length of instruction) is short; s_T will be only slightly larger than s_0. Except for very small s_0 it is unlikely to be more than two or three times s_0. Under these circumstances, the product $s_0 s_T$ is very small relative to s_T or s_0. Thus we may approximate (4) by

$$\hat{k}_1' \approx \frac{1}{T}\left[\ln\frac{s_T}{s_0}\right] \qquad (5)$$

$$\approx \frac{1}{T}\left[\ln s_T - \ln s_0\right]$$

The question is now, how does the error of estimate of k_1' depend upon T? We know from probability theory that

$$\mathrm{Var}\,\hat{k}_1' = \frac{1}{T^2}\,\mathrm{Var}\left[\ln s_T - \ln s_0\right] \qquad (6)$$

$$= \frac{1}{T^2}\left\{\mathrm{Var}\ln s_T + \mathrm{Var}\ln s_0 - 2\,\mathrm{Cov}(\ln s_T, \ln s_0)\right\}$$

If the two tests consist of different sets of questions it is unlikely that there will be any covariance of s_T and s_0. However, if there were a covariance of s_0 and s_T it would most likely be positive. In either case; zero covariance or positive covariance, we can conservatively (over-estimate) approximate Var $\hat{k}_1'$ with

$$\mathrm{Var}\,\hat{k}_1' \approx \frac{1}{T^2}\left[\mathrm{Var}\ln s_T + \mathrm{Var}\ln s_0\right] \qquad (7)$$

As an aside, if a random variable x has density f(x) with $E(x) = \mu$ and $\mathrm{Var}(x) = \sigma^2$ and we wish to calculate Var ln x, the following approximation by Taylor Series is most useful

$$\ln x \approx \ln\mu + \frac{d}{dx}\ln x \Bigg|_{\mu} \cdot (x - \mu)\,.$$

Differentiating ln x and evaluating at $x = \mu$, then taking variances on both sides of the Taylor approximation we have

$$\begin{aligned} \text{Var} \ln x &\approx \text{Var} \ln \mu + \text{Var} \frac{1}{\mu}(x-\mu) \\ &\approx 0 + \frac{1}{\mu^2} \text{Var}(x-\mu) \\ &\approx \frac{\sigma^2}{\mu^2} \end{aligned} \tag{8}$$

With this result we now return to the problem of computing the variance of the estimated k_1'.

Referring back to equation (7) and making use of the fact that s_0 is distributed with mean x_0 and variance equal to $x_0(1-x_0)/N$, or

$$E(s_0) = x_0, \quad \text{Var}(s_0) = \frac{x_0(1-x_0)}{N}$$

and for s_T as well,

$$E(s_T) = x_T, \text{Var}(s_T) = \frac{x_T(1-x_T)}{N}$$

we make use of equation (8) to conclude that

$$\begin{aligned} \text{Var}\,\hat{k}_1' &\approx \frac{1}{T^2}\left[\frac{x_T(1-x_T)}{x_T^2\ N} + \frac{x_0(1-x_0)}{x_0^2\ N}\right] \\ &\approx \frac{1}{NT^2}\left[\frac{(1-x_T)}{x_T} + \frac{(1-x_0)}{x_0}\right] \end{aligned} \tag{9}$$

Although, x_0 and x_T are unknown parameters, we see that the estimation error decreases linearly with increased test size and decreases more rapidly with longer periods of final instruction, T. This is a useful result. We obtain a more precise estimate of the parameter k_1' if the length of instruction is doubled than if we triple the number of questions on the pre and post tests. We may rewrite (9) more clearly as

$$\text{Var}\, \hat{k}_1' \approx \frac{1}{NT^2}\left(\frac{1}{x_T} + \frac{1}{x_0} - 2\right) \tag{10}$$

Generally, as T increases, x_T would also increase and so the reduction in Var $\hat{k}_1'$ is even more pronounced when T is increased.

Equation (10) enables us to make some informed choices in designing a pretesting instrument. We do not know the values of x_T and x_0, but we may use assumed or representative values (s_T and s_0) in their places. With an approximate value for the error of estimate of k_1' we are in a position to judge the reliability of the parameters used in generating schedules.

By analogy with k_1', we can estimate the variance of the estimate of k_2, the intercept. By equation (3) rewritten here we have

$$\hat{k}_2 = -\ln\left(\frac{1}{s_0} - 1\right)$$

$$= \ln s_0 - \ln(1 - s_0) \qquad (3)$$

So,

$$\mathrm{Var}\,\hat{k}_2 = \mathrm{Var}[\ln s_0] + \mathrm{Var}[\ln(1 - s_0)] - 2\,\mathrm{Cov}[\ln s_0, \ln(1 - s_0)].$$

Now Cov ($\ln s_0$, $\ln(1 - s_0)$ must be found since it is surely negative and, therefore, contributes to Var $\hat{k}_2$.

$$\mathrm{Cov}[\ln s_0 \ln(1 - s_0)] = E[\ln s_0 \ln(1 - s_0)] - E\ln s_0 E\ln(1 - s_0) \qquad (11)$$

We must do a Taylor Series expansion of the expectation of the cross product. For accuracy three terms will be used in expanding around x_0.

$$E[\ln s_0 \ln(1-s_0)] \approx \ln x_0 \ln(1-x_0) - \frac{1}{2N}\left[\frac{(1-x_0)^2\ln(1-x_0)+x_0{}^2\ln x_0}{x_0(1-x_0)} - 2\right]$$

Also $E \ln s_0 E \ln (1 - s_0)$ is given approximately by

$$E \ln s_0 E \ln(1-s_0) \approx \ln x_0 \ln(1-x_0) - \frac{(1-x_0)\ln(1-x_0)}{2Nx_0} - \frac{x_0 \ln x_0}{2N(1-x_0} + \frac{1}{4N^2} \qquad (12)$$

Thus from (11)

$$\mathrm{Cov}\left[\ln s_0 \ln(1-s_0)\right] \approx \left[\frac{4N-1}{4N^2}\right] \qquad (13)$$

From this we obtain the result

$$\mathrm{Var}\,\hat{k}_2 \approx \frac{1-2x_0}{Nx_0(1-x_0)} - \frac{4N-1}{2N^2} \qquad (14)$$

Given small values of x_0, the first term in expression (14) increases rapidly. Thus the variance of $\hat{k}_2$ grows rapidly if the student has a low expected score for the first test, x_0. If we wish to reduce this error variance, the only controllable variable is the size of the test, N. This result implies that for a given student (fixed value of x_0) the reliability of our estimate is a function of test size N, only.

Other properties of parameter Estimates $\hat{k}_1'$ and $\hat{k}_2$: Bias

Before using the estimated learning curve parameters $\hat{k}_1'$ and $\hat{k}_2$ we can check whether or not these statistics are good estimates of the underlying values (assuming Model II holds). It is useful to know whether or not the estimation procedure will produce results--on the average--that converge to the true values k_1' and k_2. Here, the test of goodness of an estimate is whether or not it is biased. As we shall see the statistics $\hat{k}_1'$ and $\hat{k}_2$ are biased estimators, that is

$$E(\hat{k}_1') \neq k_1' \text{ and } E(\hat{k}_2) \neq k_2.$$

The calculation of even an approximation to the expected value of $\hat{k}_1'$ and of $\hat{k}_2$ is a large undertaking and does not produce a result that can be analyzed readily. It is possible, however, to demonstrate that these statistics are biased estimators in the following very direct manner.

Suppose that $E(\hat{k}_2) = k_2$. Then using the notation of the preceding sections we have that

$$k_2 = E(\hat{k}_2) \equiv E(f(s_0))$$

But, by definition, $k_2 = f(x_0)$ where $x_0 = E(s_0)$

Thus, by hypothesis we have

$$E[f(s_0)] = k_2 = f(E[s_0])$$

i.e. $$E(f(s_0)) = f(E(s_0)).$$

This last equation holds if and only if the function $f(\cdot)$ is a linear function. But

$$f(s) \equiv -\ln\left(\frac{1}{s} - 1\right)$$

is notoriously non-linear. Thus we are forced to reject the original premise that $E(\hat{k}_2) = k_2$.

A similar proof by contradiction shows that $\hat{k}_1'$ is a biased estimator of k_1'. In fact, both $\hat{k}_1'$ and $\hat{k}_2$ are under-estimators of the absolute values of k_1' and k_2.

It is possible, in principle, to formulate a cost of error function (or a total estimation cost function) in terms of Var $\hat{k}_1'$ and Var $\hat{k}_2$, the estimation bias, and N and T. With constraints (or prices) on N and T it might even be possible to carry out a search (by gradient methods) on the values of N, T and the biases so that the cost function is minimized. There are many practical reasons why this ideal is impractical. One good reason is that the exponential function for x_T would make the calculations prohibitively expensive. In practice, the choice of N and T will be made heuristically, on the basis of experience.

Test of Parameter Estimation Procedure

The estimation procedure just developed was given a limited test by running it on initial portions of the student histories originally used to test Model II in Chapter Four.

The first complete topic in each student's history (simulating the selection aid) was used to estimate the values of k_1' and k_2 for the respective students. The scores s_0 and s_T were taken as the first pair of topic test scores. For each individual, T was the actual length of console time between the tests. (T varied across students.) In this way values of $\hat{k}_1'$ and $\hat{k}_2$ were calculated for 19 out of the 20 sets of data originally analyzed in Chapter Four. One student was dropped from the analysis since he apparently failed to take the second test, or his results were not recorded. The values of k_1' and k_2 have already been estimated for these students. The new estimates, obtained by working with the first pair of tests, (in place of a real on-line selection aid) were compared with the previously estimated values.

As expected from the previous theoretical discussion the selection aid estimates were consistent underestimators of the "true" values. ["True" values here refer to the estimates of k_1' and k_2 from data covering the whole course of instruction]. The selection aid underestimated the value of k_1' by an average of 8.05%. The estimates of k_2 made by the selection aid procedure were low by an average of 19.0%.

The selection aid estimates were regressed on the Chapter Four values of k_1' and k_2. An interesting result that came from this analysis is that the selection aid estimate of k_2, the intercept, was significantly correlated with the "true" value ($r=0.8960$, $n=19$; reject H_0: $\rho = 0$ at $\alpha = 0.01$). On the other hand the estimates of the slope parameter k_1', from the selection aid, did not agree well with the true values. When the two sets of estimates were regressed on one another the hypothesis of zero correlation could not be rejected. ($r = 0.3552$, $n = 19$).

The conclusion drawn from the lack of correlation between the true and selection aid values of k_1' is that T must be greater than the length of one Topic--at least on the Biology curriculum. An experimental program is needed to learn the best length of trial instruction.

SUMMARY

In this chapter we have been concerned with obtaining estimates of student learning characteristics for use in making scheduling decisions. After outlining an ideal data gathering procedure and discussing the variables of interest an experimental data gathering tool was discussed. Following these two unsuccessful attempts to find off-line predictors of Model II parameters were reported. Finally, to meet the needs of the scheduling problem an on-line selection aid has been designed and some statistical properties of its estimates were examined. A limited test of the on-line selection aid has been made and further experimentation is called for.

REFERENCES

Chapter Six

[1] Ackoff, Russell L., and Sasieni, Maurice W. Fundamentals of Operations Research, New York: John Wiley and Sons, 1968.

[2] Atkinson, Richard. "Computerized Instruction and the Learning Process." American Psychologist, Volume 23, No. 4, April 1968.

[3] Charp, Sylvia, and Hayman, John L., Jr. "The Research Design for the Philadelphia Program in Computer Assisted Instruction." School District of Philadelphia Division of Instructional Systems, Mineographed, 1967.

[4] Engel, Gerald L. "Computer Assisted Instruction: A Selected Bibliography and KWIC Index." Dahlgren, Virginia: U. S. Naval Weapons Laboratory, Report No. K-3/67, January 1967.

[5] Gagné, R. M. The Conditions of Learning. New York: Holt Rinehart and Winston, Inc., 1965

[6] Galanter, E. H., (Ed.) Automatic Teaching: The State of the Art. New York: John Wiley and Sons, 1959.

[7] Gentile, J. Ronald. "The First Generation of Computer Assisted Instruction Systems: An Evaluative Review." A.V. Communications Review, Volume 15, No. 1, 1967, pp. 23 - 53.

[8] Glassman, Jerrold. Programmed Reading. New York: Globe Book Co., 1966.

[9] Groen, G. J., and Atkinson, R. C. "Models for Optimizing the Learning Process." Technical Report No. 92, February 11, 1966, Institute for Mathematical Studies in The Social Sciences, Stanford University, Stanford, California.

[10] Hickey, Albert E. "Computer Assisted Instruction: A Survey of the Literature." 2nd Edition, Newbury Port, Massachusetts: Entelek, Inc., 1967.

[11] Hilgard, Ernest R. Theories of Learning. New York: Appleton Century Crofts, second edition, 1956.

[12] Hughes, J. L. Programmed Instruction for Schools and Industry. Chicago: Science Research Associates (pub.), 1962.

[13] Kagan, Jerome. "Reflection - Impulsivity and Reading Ability in Primary Grade Children." Child Development, Volume 36, September 1965.

[14] Kopstein, Felix F., and Seidel, Robert J. "Computer Administered Instruction Versus Traditionally Administered Instruction: Economics." A.V. Communications Review, Volume 16, No. 2., Summer 1968, pp. 147-175.

[15] Roe, Arnold. "An Adaptive Decision Structure for Educational Systems." Los Angeles: Report No. 63 - 63, Department of Engineering, University of California, Los Angeles, 1962.

[16] Scheflen, Albert E. "Human Communication: Behavioral Programs and their Integration in Interaction." Behavioral Science, Volume 13, No. 1, January 1968.

[17] Schwartz, H. A., and Long, H. S. "Instruction by Computer." Datamation, September 1966, pp. 73-79.

[18] Seidel, Robert J., and Kopstein, Felix F. "A General Systems Approach to the Development and Maintenance of Optimal Learning Conditions." Alexandria, Virginia: George Washington University, Human Resources Research Office, September 1967.

[19] Silvern, Gloria M., and Silvern, Leonard C. "A Graduate Level University Course in Methods of Computer Assisted Instruction." I. E. E. E. Transactions on Human Factors in Electronics, Volume HFE-8 No. 2, June 1967, pp. 94 - 102.

[20] Skinner, B. F., and Holland, James G. "The Use of Teaching Machines in College Instruction." In Teaching Machines and Programmed Instruction, Lumsdaine and Glaser (ed.), 1960.

[21] ______. The Technology of Teaching. New York: Appleton Century Crofts, Division of Meredith Corporation, 1968.

[22] Smallwood, Richard D. A Decision Structure for Teaching Machines. Cambridge, Massachusetts: The M. I. T. Press, 1962.

[23] ______; Weinstein, Iram; and Eckles, James E. "Final Report on Office of Naval Research Contract Nonr-225(84)." March 15, 1967, Institute in Engineering - Economic Systems, Stanford University, Stanford, California.

[24] Stolurow, L. M. Teaching by Machine. Washington, D. C.: U. S. O. E. Monograph No. 6, U. S. Government Printing Office, 1961.

[25] ______. "Systems Approach to Instruction." University of Illinois Training Research Laboratory Technical Report No. 7, 1965.

[26] ______. "Computer-Based Instruction: Psychological Aspects and Systems Conception of Instruction." Report on ONR Contract N00014-67-A-298-0003, 1967.

[27] Swets, John A.; Harris, Judith R.; McElroy, Linda S.; and Rudloe, Harry. "Computer Aided Instruction in Perceptual Identification." Behavioral Science, Volume 11, No. 2, March 1966, pp. 98 - 104.

[28] Webb, Eugene; Campbell, Donald T.; Schwartz, Richard D.; and Sechrest, Lee. Unobtrusive Measures: Nonreactive Research in the Social Sciences. Chicago: Rand-McNally & Co., 1966.

[29] Wilson, H. A., and Atkinson, R. C. "Computer Based Instruction in Initial Reading: A Progress Report on the Stanford Project." Technical Report No. 119, August 25, 1967, Institute for Mathematical Studies in the Social Sciences, Stanford University, Stanford, California.

CHAPTER SEVEN

THE SCHEDULING SYSTEM

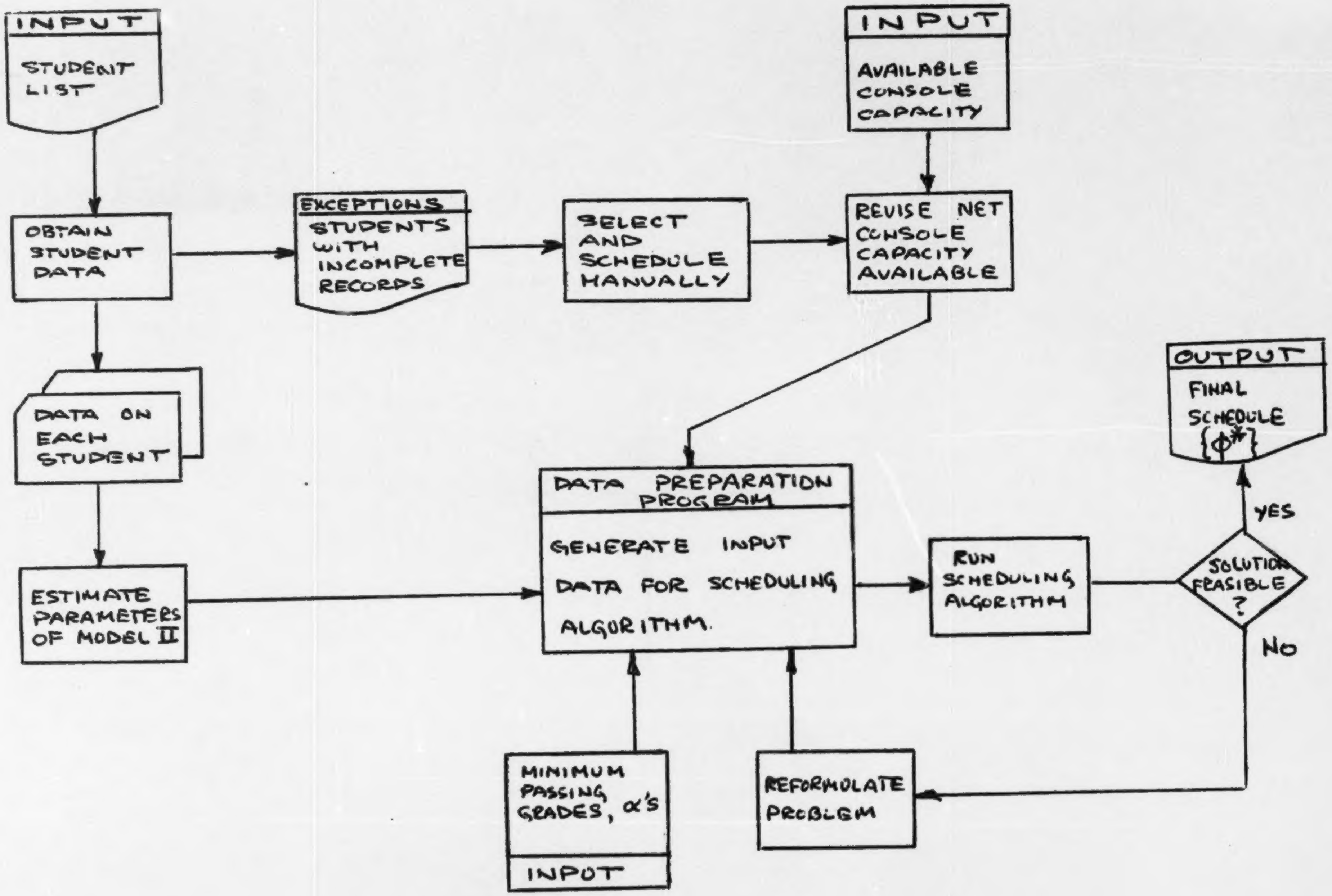
INTRODUCTION

This chapter organizes the previously formulated scheduling problem and estimation procedures into a unified system. Although parts of this system are automatic, some manual operations are required to prepare data and to interpret intermediate results. A case is presented to illustrate the use of the procedure in generating a schedule.

OVERVIEW OF THE SCHEDULING PROCEDURE

Figure 7.1 below is a schematic overview of the student scheduling procedure. We assume that a list of the students who are eligible for CAI, or who have requested it, and an estimate of available console capacity can be obtained. Most of the activities shown on the diagram are data processing support for the separable programming algorithm which solves the scheduling problem formulated in Chapter Five.

Data must be gathered for eligible students by assigning them to a trial console period on a sample of curriculum. The names of


INPUT
STUDENT LIST
OBTAIN STUDENT DATA
EXCEPTIONS
STUDENTS WITH INCOMPLETE RECORDS
SELECT AND SCHEDULE MANUALLY
INPUT
AVAILABLE CONSOLE CAPACITY
REVISE NET CONSOLE CAPACITY AVAILABLE
DATA ON EACH STUDENT
ESTIMATE PARAMETERS OF MODEL II
DATA PREPARATION PROGRAM
GENERATE INPUT DATA FOR SCHEDULING ALGORITHM.
RUN SCHEDULING ALGORITHM
SOLUTION FEASIBLE ?
YES
NO
OUTPUT
FINAL SCHEDULE {φ*}
MINIMUM PASSING GRADES, α's
INPUT
REFORMULATE PROBLEM

Schematic of Student Scheduling Procedure

Figure 7.1

students who have missing or incomplete records are put onto an exception list for special processing. The number of students specially scheduled because of lacking data will vary from school to school. It may be necessary to use random selection and scheduling of these exceptions. In the long run (as CAI is better supported) the data needed in managing CAI will be routinely gathered and exceptions should be a minor problem.

Estimates are made of the model parameters k_1' and k_2 for each student. These parameters, together with console capacity information, failure probabilities, and course goals for the achievement test, are input to the data preparation routine. The output of this routine is a deck of punched cards which enable the separable programming algorithm to solve the scheduling problem described in (equation 5.9).

$$\left.\begin{array}{llll} \text{Find} \quad \max_{\{\Phi\}} \sum_{i=1}^{n} x_i(\Phi_i) & & & (5.2a) \\ \text{subject to the constraints} & & & \\ \Phi_i \ge \frac{1}{k'_{2,i}}\left[\ln\left(\frac{\pi_i}{1-\pi_i}\right) - k_{2,i}\right] & & i=1,---,n & (5.8) \\ \sum_{i=1}^{n} \Phi_i \le c & & & (5.1) \\ \Phi_i \ge 0 & & i=1,---,n & \end{array}\right\} (5.9)$$

In solving this problem we will replace 5.8 with 5.6 (i.e. $x_i \ge \pi_i$).

Depending upon the imbalance between needs and resources the scheduling algorithm may or may not produce a feasible schedule. If there is a feasible optimum, the final class schedules are produced. In the event that needs are greater than available resources, an analysis of the infeasible problem and a reformulation are required. The scheduling procedure is complete when final class assignments are generated by a modified Northwest corner procedure.

The remaining sections of this chapter highlight important details of the scheduling procedure.

SEPARABLE PROGRAMMING: IMPLEMENTING THE SCHEDULING ALGORITHM

Chapter Five showed that the scheduling problem (equation 5.9) could be solved by separable programming. Once the learning model parameters are estimated for each pupil, the data must be prepared to conform to the requirements of the separable programming algorithm. A computer program has been written to organize the data for processing by the IBM Mathematical Programming System (MPS/360) [3,4].

Description of Separable Programming

The scheduling problem equation 5.9 is solved by applying the delta-method of separable programming presented by Hadley [2]. Its derivation will not be repeated here. Non-linear functions (each involving only one variable) appear in both the constraint set and objective of 5.9. All can be approximated (as closely as necessary) in the so called piecewise linear manner. In mathematical terms these approximations are often called a polygonal approximation. The piecewise linear functions become the basis for finding the optimal schedule $\{\Phi^*\}$ of 5.9 by applying a modified simplex algorithm.

The separable programming adds a set of logical restrictions to the standard simplex algorithm. These restrictions together with the polygonal approximations of the original non-linear problem cause the simplex algorithm to optimize the non-linear functions. The logical restrictions on the simplex algorithm are limitations on the order in which variables enter the basic feasible solution. Figure 7.2 illustrates these restrictions.

Let $0 \le x1 \le 1$ when $0 \le x \le d_1$

$0 \le x2 \le 1$ when $d_1 \le x \le d_1 + d_2$

$0 \le x3 \le 1$ when $d_1 + d_2 \le x \le d_1 + d_2 + d_3$

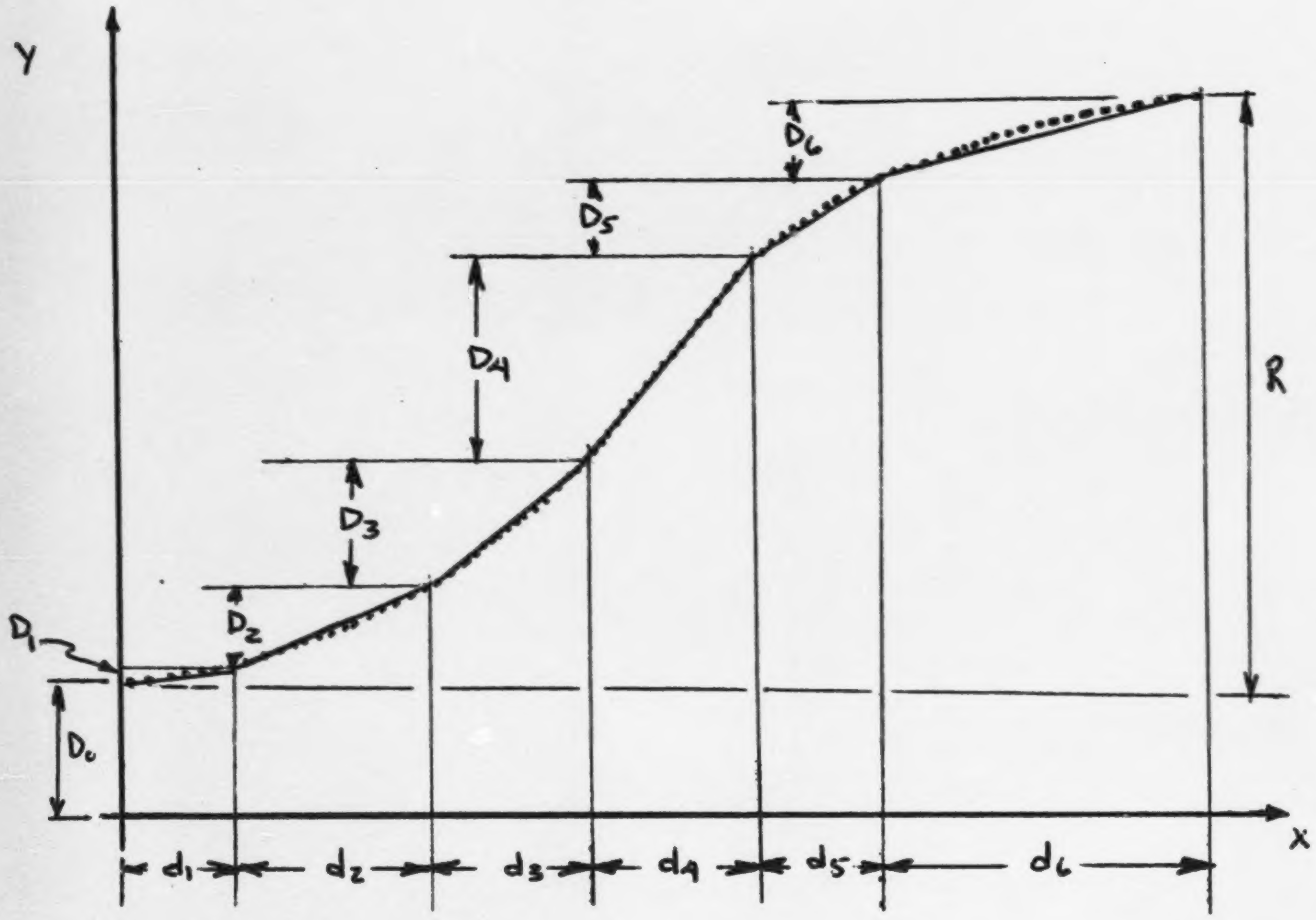
and so on.

In the interval $0 \le x \le d_1$ we approximate the function $y = f(x)$ by

$$y = D_0 + D_1 x1$$

Over the second interval $d_1 \le x \le d_1 + d_2$ the function is approximated by

$$y = D_0 + D_1 x1 + D_2 x_2 \text{ with } x1 = 1\,; \quad 0 \le x2 \le 1$$


Y
X
D₀
D₁
D₂
D₃
D₄
D₅
D₆
R
d₁
d₂
d₃
d₄
d₅
d₆

PIECEWISE LINEAR APPROXIMATION TO A NONLINEAR CURVE
FIGURE 7.2.

On the third interval $d_1 + d_2 \le x \le d_1 + d_2 + d_3$ the approximation is

$$y = D_0 + D_1 x1 + D_2 x2 + D_3 x3 \text{ where } x1 = x2 = 1; \quad 0 \le x3 \le 1.$$

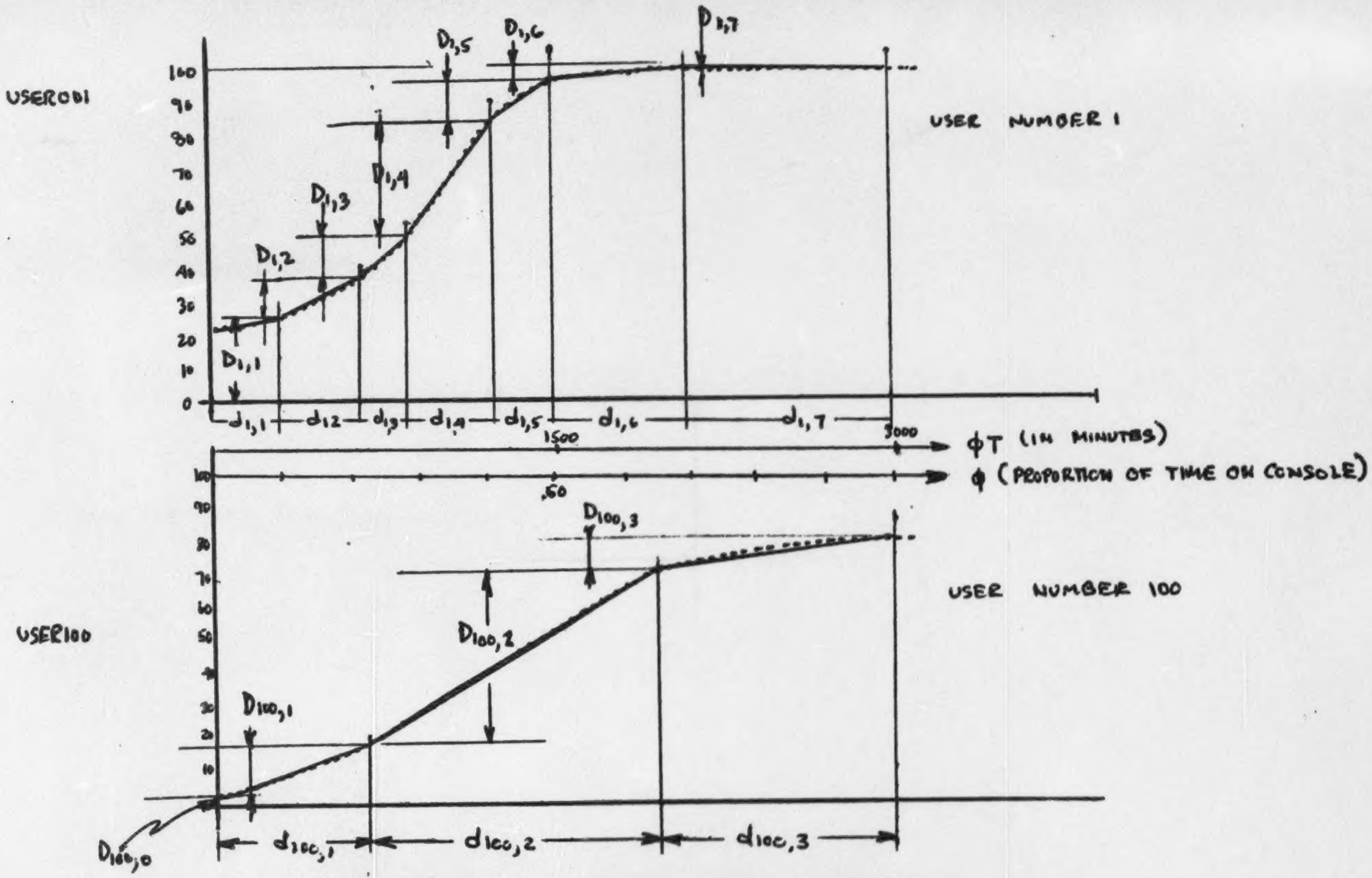
The variables used in the piecewise approximation are called restricted variables. To approximate the function over the N-th interval all preceding restricted variables (x1, x2, ---, xN-1) must equal 1, while all higher variables xn+1, xN+2, etc. are identically zero. Thus the simplex algorithm is constrained from changing x3, say, unless x2 and x1 are already at their upper bounds (x1 = x2 = 1). In practice this is accomplished by not allowing x3 to enter a basic solution unless x1 and x2 are already at their upper bounds. This is the origin of the term "restricted basis entry".

Data Preparation Program

Often the approximations to the non-linear functions in a problem are made by hand. This is far too laborious for the student scheduling process. A program was developed to automatically calculate the piecewise approximations of the learning models and to convert them to the specially coded information required by the MPS/360 algorithm. The process starts by computing a maximum allowable error of approximation. If very little error is allowed, there will have to be many line segments used in approximating even a smooth curve. The size of the ultimate problem is affected dramatically by the number of segments, since every segment requires one restricted variable (and therefore a column in the data matrix). In Figure 7.2 the error criterion is a constant times the dimension R. Different multiples were tried and a value of 0.01 R produced approximations of typical functions with seven or fewer breaks.

The method of approximation is very simple. First the program approximates the function by a straight line from the function's y intercept to the point where the function meets the upper limit on x. Then at fixed intervals of x the absolute differences between the function and the approximation are computed. If the largest of these differences exceeds the allowable error the program breaks the function at the point where the largest error appears. In this way successive polygonal approximations converge very rapidly to the curves of the function. Obviously, the radius of curvature of the function and the allowable error of approximation determine the number of restricted variables needed for an adequate approximation to a given student's learning model. Figure 7.3 shows two illustrative cases, User #1 has his curve approximated by seven segments while user #100 has only three.

The dotted figure is the user's estimated learning curve (based upon k_1' and k_2). The independent variable is minutes of console time (convertible into proportion of course time on console, ϕ). The d's are the ranges on the successive restricted variables, and the D's



COEFFICIENTS FOR SAMPLE PROGRAM
ILLUSTRATION OF LINEAR APPROXIMIZATION ROUTINE
FOR TWO HYPOTHETICAL USERS.

FIGURE 7.3

supply the information required for approximating the learning curve. In Figure 7.4 we have detailed the quantities which are calculated for input to the scheduling algorithm. The data preparation program computes both the D's and the d's and punches them onto IBM cards in the MPS/360 format.

The same program which forms the linear approximations also computes the quantities π_i (as defined in Chapter Five) where $i=1,---,n$; n equals the number of students being considered for CAI. To compute these quantities the program uses a specified failure probability, the lowest passing grade and the parameters k_1' and k_2 for each student. The MPS/360 data preparation program and a printed sample of its output is listed in Appendix 7.1.

MPS/360 Data Matrix

Figure 7.4 is a sample data matrix for a 100 student scheduling problem. This example illustrates the organization of the data prepared by the preceding computer program. This data will be used below to illustrate the MPS/360 Separable Programming algorithm to solve the scheduling problem.

For 100 students, there are 101 rows and in practice there would be an average of 6 columns for all but one of the rows. Thus the data matrix is approximately 101 rows by 600 columns. Each column, including the objective function, is named. The objective function,

	U1X1	U1X2	U1X3	U1X7	U2X1	U2X2	U2X3	U2X6	....	..	U100X1	...	U100X3		RHS1	CHCOL1	CHCOL2
LINOBJ	$D_{1,1}$	$D_{1,2}$	$D_{1,3}$	$D_{1,7}$	$D_{2,1}$	$D_{2,2}$	$D_{2,3}$	$D_{2,6}$			$D_{100,1}$		$D_{100,3}$	N			
USER1	$D_{1,1}$	$D_{1,2}$	$D_{1,3}$	$D_{1,7}$	0	0	0	0			0		0	$\geq$	π_1		$-a_1$
USER2	0	0	0	0	$D_{2,1}$	$D_{2,2}$	$D_{2,3}$	$D_{2,6}$			0		0	$\geq$	π_2		$-a_2$
USER3	0	0	0	0	0	0	0				0		0	$\geq$	π_3		$-a_3$
USER4	0	0	0											$\geq$	π_4		$-a_4$
USER100	0	0	0	0	0	0	0	0			$D_{100,1}$		$D_{100,3}$	$\geq$	π_{100}		$-a_{100}$
CONSOLES	$d_{1,1}$	$d_{1,2}$	$d_{1,3}$	$d_{1,7}$	$d_{2,1}$	$d_{2,2}$	$d_{2,3}$	$d_{2,6}$			$d_{100,1}$		$d_{100,3}$	$\leq$	C	1.0	
UPBOUNDS	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0			1.0		1.0				

MATRIX OF COEFFICIENTS FOR SAMPLE PROBLEM
MATHEMATICAL PROGRAMMING SYSTEM/360 SEPARABLE PROGRAMMING DATA MATRIX

FIGURE 7.4.

top row, is named LINOBJ. Since the objective is the sum of all students' test scores, every one of the restricted variables enters into it with a weight D(u, i); u is the user number and i is the number of the interval of approximation. (These coefficients are the vertical increments calculated in the preceding computer program--see Figure 7.3).

Beneath the objective function the next rows are the individual constraints on the chance of failure. Each constraint is named; USER1, USER2, and so on. Any student's (USER) constraint includes only the piecewise approximation to his learning curve. The direction of the inequality is given as ≥, and the elements of the right hand side are $\pi_1, ---, \pi_{100}$. (See Chapter Five, equation 5.6).

The row after the USER100 row is the capacity restriction (called CONSOLES). This row includes every restricted variable for every user. Each variable has a coefficient of du, i; where u is the user number and i is the number of the interval of approximation. These coefficients are the horizontal increments (in units of π rather than in units of minutes) that have been calculated in the computer program of the preceding section--see Figure 7.3 .

The bottom row of the matrix (labeled UP Bounds) specifies the largest value that each restricted variable may assume. In this case all restricted variables have their upper bounds equal to 1.0.

Before further discussion the new terms are summarized. Each user's learning curve is approximated by several restricted variables X1, X2,---,X7 (assuming in this case that there are only 7 restricted variables). In order to distinguish one user's restricted variables from another the variable names are prefixed by U and then the user's number. Thus, U27X03 is the name of the third restricted variable of user 27's learning curve.

If we could go to the row labeled USER27 we would find all zero's except in the columns RHS1 and U27X01, U27X02,---, U27X07. In these seven columns the entries will be $D_{27,1}, D_{27,2}, ---, D_{27,7}$; all are calculated by the data preparation program.

Suppose we run the algorithm and U27X01=U27X02=U27X03=1 while U27X04=0.75 and by definition U27X05=U27X06=U27X07=0.0, in the final solution USER27's predicted achievement score would be

$$\sum_{i=1}^{7} (U27X0i) \cdot (D27,i).$$

This is given by

$$1. D27,1 + 1. D27,2 + 1. D27,3 + (0.75) \cdot D27,4 + 0$$

The amount of console capacity consumed is similarly

$$d27,1 + d27,2 + d27,3 + (0.75) \cdot d27,4 + 0.$$

This would appear in the row labeled CONSOLES, that is, in the capacity constraint.

Referring back to the matrix of coefficients, Figure 7.4 there are two added columns shown, CHCOL1 and CHCOL2. These are column vectors of change values used in parametric studies of the right hand side. Change column 1 if it is added to column RHS1 will increase console capacity by 1. Likewise, if CHCOL2 were added to RHS1 it would (for positive $a_1, ---, a_{100}$) permit us to study the effect of corresponding changes in passing grades on the optimal solution. With other CHCOL vectors we could examine the effects of other changes in the π values on the optimal solution $\{\pi^*\}$. MPS/360 can be used to perform several types of sensitivity analyses after the optimal solution is found. These procedures are detailed in the relevant IBM bulletins [3, 4].

Appendix 7.2 is a listing of an MPS/360 control program and an explanation of its main features.

INFEASIBLE SOLUTION

In scheduling students onto consoles there is a chance that no feasible solution may exist. Infeasibility may result for a variety of reasons such as too little console time available, too many students, too high a passing grade k, or unfavorable student learning characteristics. There are a wide range of possible responses available to resolve infeasibility. The decision on how to respond can only be made on a heuristic basis and we discuss it only in general terms. When infeasibility appears we may either (1) increase resources (console capacity), (2) change objectives (e.g., passing grade), or (3) both.

There are several obvious ways to change objectives. We may (1) lower the passing grade k, (2) increase α, the allowable failure probability, (3) increase the failure probability for some students only, or (4) drop some students from consideration for CAI altogether. After studying these possibilities the decision-maker may be able to define a way to cut back claims on console capacity enough so that needs balance resources. In some situations this type of analysis may provide a powerful case for obtaining more resources.

The separable programming algorithm helps in this analyzing proposed changes, since the extent of infeasibility in the original problem can be determined from detailed reports supplied automatically by the mathematical programming system. The capabilities for performing sensitivity analyses on the right hand side of the programming matrix (the constraint vector) are also very useful in deciding how to proceed in the case of infeasibility.

To use parametric sensitivity studies we have to start with a feasible solution, and it might be necessary to assume console capacity higher than what is actually available. The algorithm can be set up to reduce the console capacity to the actual figure. We then can see the effects on class achievement as the capacity is stepped down from the assumed value to the actual. Some experience with parametric studies on the MPS/360 indicates that a great deal of information can be obtained in this way at very little added cost of computing.

It is possible, even after analysis of the infeasible problem at hand, that the decision-maker may develop two or more feasible reformulations. It is likely that each reformulation will have some desirable characteristics and fall short on others. If there is no clearly dominant alternative then the problem reduces to finding the decision-maker's preferences over these possible reformulations. Very recently Stankard, Maier-Rothe and Gupta [5] have developed a general procedure for solving this type of problem.

The procedure involves asking a series of questions of the decision-maker and using linear programming to analyse the information contained in the answers. The questions are formulated so that the information obtained is consistent. By analyzing the dual linear program further questions can be asked; their answers yield a type of steepest ascent to a choice that is consistent with the decision-maker's expressed preferences. The procedure would represent a last resort in this case since it requires sequential solutions of linear programs by the simplex algorithm.

One final comment; if resources are insufficient to satisfy even the basic requirements of the students then the problem becomes at least partly one of how to acquire more resources. In the following we assume that a feasible (and optimal) schedule $\{\pi^*\}$ has been obtained. That is, we have allocated the resources at hand in such a way that the objective function is optimized subject to all expressed constraints.

FINAL CLASS ASSIGNMENT PROCEDURE

Given the optimal schedule $\{\pi^*\}$ we would like a simple and effective way to assign scheduled students to consoles. Many methods might be developed; however, the suggestion below has the advantage of simplicity. An example illustrates the development of this procedure.

Example: A biology class that meets every day of the week for one hour has two consoles available to it. There are twenty students in the class and the course lasts for ten weeks. With five days per week there will be 50 meetings of the class. In the following illustrative exercise we assign the twenty biology students to consoles so that the following (hypothetical) optimal schedule is met.

Student (i)	Schedule (Φ^*)	Console Hrs. $50\ \Phi_i^*$
1	0.20	10
2	0.40	20
3	0.30	15
4	0.20	10
5	0.20	10
6	0.30	15
7	0.25	12
8	0.15	8
9 - 20	-0-	0

Sample Class Schedule

Figure 7.5

The daily console assignments will be made by a modified Northwest corner procedure [1]. A tableau, such as Figure 7.6 below, summarizes the resources available and the students scheduled requirements. There is a row for each class meeting day of the week; or when considering several periods per day there would be a row for each period in every day. Each student scheduled for console time is represented by a column. Note that the students 9-20 are treated as one--they have the same requirements--for no console time.

FOR ONE PERIOD

DAY \ i	1	2	3	4	5	6	7	8	9-20	
Mon										20
Tues										20
Wed										20
Thurs										20
Fri										20
	10	20	15	10	10	15	12	8	0	

Assignment Tableau for One Week

Figure 7. 6

The number of console hours available during each of the days of the week is written in the corresponding cell of the right hand side of the tableau. Looking at the first row, for example, the 20 at the right is the number of console hours available on each Monday that the course meets. Thus, 2 consoles x 10 weeks of classes on Monday results in the 20. The bottom of each student's column shows the number of periods of instruction (out of 50) scheduled for him in the optimal schedule.

The assignment of students to console periods proceeds in a mechanical fashion once the row and column totals are entered. Start at the upper left hand corner and find the smaller of the row or column totals. For the MON row and student 1, the column total 10 is smaller than the row total, 20. Enter this quantity (10) in that cell, subtract it from both row and column totals and strike out any total which is reduced to zero, (in this case the column 1 total is struck). Looking at tableau 2 in Figure 7.7 below, we have just assigned student 1 (who required 10 hours of console time) to a console for each Monday that the course meets. There are still 10 hours left on Monday (the new right hand row total for MON) so we move right

DAY \ i	1	2	3	4	5	6	7	8	9-20		
Mon	10	10								~~20~~	~~10~~
Tues		10	10							~~20~~	~~10~~
Wed			5	10	5					~~20~~	~~15~~ ~~5~~
Thurs					5	15				~~20~~	15
Fri										20	
	~~10~~	~~20~~	~~15~~	~~10~~	~~10~~	15	12	8	0		
		~~10~~	~~5~~		~~5~~						

Example Tableau 2

Figure 7.7

to the second column, first row. The new row total, 10, is smaller than the column 2 total, 20, so it is entered in the cell (MON, Student 2).

The value 10 is subtracted from both row and column totals and the row total for MON is struck out since it is now zero.

There is no console time left on Monday, yet student 2's schedule requirements are not satisfied. We are, therefore, forced to move down to the TUESDAY row. Again, enter the smaller of the row or column totals--10 from the remaining column total--into the (TUESDAY, Student 2) cell. Subtract 10 from both row and column totals here and cross out the column 2 total, since student #2's total requirements of 20 hours are satisfied. The usual Northwest corner procedure (moving from the top left or Northwest corner--as in the preceding) is carried out until we arrive at student #6 on THURS. This entry has been circled in tableau 2, Figure 7.7.

The circled value, 15, in cell (THURS, Student 6) is impossible to implement; it resulted from applying the Northwest corner rule and so the rule must be modified. The entry requires student 6 to have 15 Thursday console sessions; in a course that meets one period per day for ten weeks this is impossible. The modification that we must make to the procedure is; if too large an entry is allowed by the rule, continue working down the column in question until the column total is reduced to zero, then move right to next column and go up to the topmost row that still has a non zero total on the right hand side.

Obviously the largest possible entry in any cell for this case, is 10. We can only put the student on one console for each of the meetings of the class that day. In practice this maximum value is determined before we start the assignment process and it is checked before each entry is made.

This restriction has been applied in producing tableau 3, Figure 7.8 below. The arrows show the moves required by the restriction to eliminate the infeasible assignment.

DAY \ i	1	2	3	4	5	6	7	8	9-20		
Mon	10	10								~~20~~	~~10~~
Tues		10	10							~~20~~	~~10~~
Wed			5	10	5					~~20~~	~~15~~ ~~5~~
Thurs					5	10 ↓	↗ 5 ↓			~~20~~	~~15~~ ~~5~~
Fri						5 ↗	7	8		~~20~~	~~15~~ ~~5~~
	~~10~~	~~20~~	~~15~~	~~10~~	~~10~~	~~15~~	~~12~~	8	0		
		~~10~~	~~5~~		~~5~~	~~5~~	~~7~~				

Example Tableau 3

Figure 7.8

The final assignment--tableau 3, Figure 7.8, is easy to interpret. Looking at student 6, for instance, he has 10 hours of time on Thursday and 5 hours on Friday. So, for example, he might spend every Thursday and every other Friday on the console (for 10 out of 10 and 5 out of 10 hours respectively). The teacher could make the final decisions about student 6's Friday console session because they will affect the assigned Friday sessions for students 7 and 8.

APPENDIX 7.1

CHAPTER SEVEN

DATA PREPARATION PROGRAM

The main element of this appendix is the accompanying FORTRAN program listing. The discussion here will be limited to a brief commentary on the main features of that program.

The program requires the parameters of the learning model for each student. It uses the procedure described in this chapter to generate piecewise approximations to the functions, and then it punches a deck of cards that are used as input to the MPS/360 separable programming algorithm.

Main Program

Lines	Function
0001-0015	Defines variables, reads the formats of the MPS/360 right hand side cards, reads the number of users, inputs the k_1', k_2 and $Z\alpha$ for each user.
0016	Reads the achievement test passing grades and the number of test questions on that test.
0017	Repeats the approximation process for each student user.
0018-0021	Calculates the value of $x(\Phi)$, for the current user, at $\Phi = 0.0,\ 0.05,\ 0.10, ---, 1.0$. This assumes that the next time available for the

Lines	Function
0018-0021 (Cont'd)	course is 2150 minutes (approximately 12 weeks x 5 classes per week x 36 minutes, net per period.)
0022	Establishes the maximum error of approximation as (0.01) x [x(1.0) - x(0)].
0023-0025	Computes each student's minimum passing grade.
0026-0029	Initializes BREAK, a vector whose elements will be the numbers of the points where the function has been broken.
0030-0044	Computes the value of the approximation at each point on the Φ axis.
0045-0048	Computes the approximization error at each point on the Φ axis.
0048-0061	Sorts all the errors by absolute magnitude and sets IBIG equal to the value of Φ where the maximum error, greater than ALLOW, occurs.
0062-0073	Revises the list of points where the function is broken and returns to the point at which the values of the approximating function are computed--line 0030
0074-0100	Uses the final approximation to the user's curve to print out and punch the control cards for the columns, rows and bounds sections of the MPS/360 input deck.
0101-0106	Writes out the right hand side control cards for the MPS/360 input deck.
0107-0109	End of Main Program

Function RHSF(PASSK, PI, NTESTQ)

Lines	Function
0001-0003	Converts number of test questions NTESTQ and $Z\alpha$= (PI) into forms used in computing the passing grade constraint.
0004-0006	Computes the terms in equation 5. 6.
0007	Calculates the lower bound on the expected score that satisfies the chance constraint.
0008-0009	End of function.

Function F is the Fortran command for Model II.

The remaining lines are first the input to the data preparation program and then a sample of the output of this program. The punched cards produced by this program have images that are identical to the printed output. The numerical values shown were calculated from the actual learning curves that were fit to the data from students in the Germantown High School Biology Spring, 1968. The values of k_1 and k_2 were taken from Table 4. 1. The value of N is 40 and $Z\alpha = 0.0$ with a passing grade $k \doteq 50\%$.

```
FORTRAN IV G LEVEL 1, MOD 3           MAIN           DATE = 69034         23/16/11

                C      THIS PROGRAM PREPARES INPUT DATA TO THE MPS360 SEPARABLE PROGRAMMING
                C      PACKAGE.
                C      *********CHANCE CONSTRAINED PROGRAMMING FORMULATION*****
 0001                  READ(5,900) NUMUSE
 0002              900 FORMAT(I3)
 0003                  REAL FUNC(20),FKONE(100),FKTWO(100),PI(100),APPROX(20),ERROR(20)
 0004                  REAL PHSPI(100)
 0005                  INTEGER TOTERR
 0006                  INTEGER BREAK(20)
 0007                  REAL    VF1(20),VF2(20),VF3(20),VF4(20)
 0008                  READ(5,915) (VF1(K), K = 1,20)
 0009                  READ(5,915) (VF2(K), K = 1,20)
 0010              915 FORMAT(20A4)
 0011              101 CONTINUE
 0012                  DO  1 I=1,NUMUSE
 0013                  READ(5,901) FKONE(I),FKTWO(I),PI(I)
 0014                1 CONTINUE
 0015              901 FORMAT(4F8.3)
 0016                  READ(5,901) PASSK,NTESTO
 0017                  DO 4 IUSER  = 1,NUMUSE
 0018                2 DO 3 INTRVL = 1,20
 0019                  T = 2150.
 0020                  PHI =FLOAT(INTRVL)/(20.0)
 0021                3 FUNC(INTRVL)= F(PHI,FKONE(IUSER),FKTWO(IUSER),T)
 0022                  ALLOW = (.01  )*(FUNC(20) - FUNC(1) )
 0023                  DO 40 I = 1, NUMUSE
 0024                  PHSPI(I) = PHSF(PASSK,PI(I),NTESTO)
 0025               40 CONTINUE
 0026                  DO 5 I=1,20
 0027                5 BREAK(I)=20
 0028                  BREAK(1)=1
 0029                  BREAK(2)=20
 0030                6 DO 9 I=1,19
 0031                  I1=1
 0032                7 IF(BREAK(I1).LE.  I .AND. I .LE. BREAK(I1+1)) GO TO 8
 0033                  I1 =I1 + 1
 0034                  GO TO 7
 0035                8 CONTINUE
 0036                  IF(BREAK(I1+1) .EQ. BREAK(I1) ) GO TO 108
 0037                  TOP = FLOAT(I- BREAK(I1))
 0038                  RUN     = FLOAT(BREAK(I1+1)-BREAK(I1)   )
 0039                  RISE = FUNC(BREAK(I1+1)) - FUNC(BREAK(I1))
 0040                  APPROX(I)= FUNC(BREAK(I1)) +(TOP/RUN)*RISE
 0041                  GO TO 9
 0042              108 CONTINUE
 0043                  APPROX(I) = FUNC(BREAK(I1) )
 0044                9 CONTINUE
 0045                  DO 10 I = 1,19
 0046                  ERROR(I) = ABS( FUNC(I)-APPROX(I))
 0047               10 CONTINUE
 0048                  ERROR(20)=0.0
                C      FIND THE LARGEST ERROR GREATER THAN ALLOW AND BREAK FUNC THERE.
 0049                  IBIG = 1
 0050                  INT = 1
 0051                  TOTERR = 1
 0052               11 IF(ERROR(INT).GT.ALLOW) GO TO 12
 0053                  INT = INT + 1
 0054                  TOTERR = TOTERR + 1
```

```
FORTRAN IV G LEVEL 1, MOD 3            MAIN             DATE = 69034        23/16/11

 0055                 IF(TOTERR .EQ. 19) GO TO 18
                C     THE APPROXIMASTION PROCEDURE TERMINATES WHEN ALL ERRORS ARE LESS THAN
                C     ALLOW
 0056                 IF(INT.LT. 20) GO TO 11
 0057                 IF(INT .EQ. 20) GO TO 13
 0058              12 IF(ERROR(INT).GT.ERROR(IBIG)) IBIG =INT
 0059                 INT = INT + 1
 0060                 IF(INT .LT. 20) GO TO 11
 0061              13 CONTINUE
                C     IBIG IS THE INDEX FOR WHICH (ERROR) IS MAX.  THE FUNCTION IS BROKEN HERE
 0062                 I = 1
 0063              14 IF(BREAK(I) .LT. IBIG  .AND. IBIG .LT. BREAK(I+1)) GO TO 15
 0064                 I = I + 1
 0065                 IF (I .GT. 18) GO TO 116
 0066                 IF (I .LE. 20) GO TO 14
 0067              15 IP1 = I +1
 0068                 I2 = 20 - IP1
 0069                 DO 16 I1 = 1,I2
 0070              16 BREAK(21-I1) = BREAK(20 -I1)
 0071             116 CONTINUE
 0072                 BREAK(IP1)= IBIG
                C     IF THIS APPROXIMATION HAS NO ERRORS > ALLOW THE PROGRAM JUMPS TO 18
                C     OTHERWISE IT RETURNS TO THE BEGINNING OF THE PROCESS, STATEMENT 6
 0073              17 GO TO 6
 0074              18 CONTINUE
 0075                 IBREAK =1
 0076              19 CONTINUE
 0077                 IARG2 = BREAK(IBREAK + 1)
 0078                 IARG1 = BREAK(IBREAK)
 0079                 OBJLIN = FUNC(IARG2) - FUNC(IARG1)
 0080                 IZERO = 0
 0081                 IF(IBREAK .EQ. 1) OBJLIN=F(IZERO,FKONE(IUSER),FKTWO(IUSER),T) +
                     1  FUNC(IARG2) - FUNC(IARG1)
 0082                 DO 21 NNUNIT = 6,7
                C     IF (NNUNIT .EQ. 7) GO TO 21
 0083                 DELX = (FLOAT(IARG2 - IARG1)) / 20.
 0084                 IF(IUSER .LT.10) WRITE(NNUNIT,902) IUSER,IBREAK,OBJLIN,IUSER,DELX
 0085             902 FORMAT(4X,'U00',I1,'X0',I1,3X,'LINOBJ',T25,F8.3,T40,'PHI00',I1,T50
                     1,F8.3)
 0086                 IF(IUSER .GE.10) WRITE(NNUNIT,903) IUSER,IBREAK,OBJLIN,IUSER,DELX
 0087             903 FORMAT(4X,'U0',I2,'X0',I1,3X,'LINOBJ',T25,F8.3,T40,'PHI0',I2,T50,
                     1F8.3)
 0088                 IF(IUSER.LT.10)WRITE(NNUNIT,904)IUSER,IBREAK,IUSER,OBJLIN,DELX
 0089             904 FORMAT(4X,'U00',I1,'X0',I1,3X,'USER00',I1,T25,F9.4, T40,'CONSOLE',
                     1T50,F9.4 )
 0090                 IF(IUSER.GE.10)WRITE(NNUNIT,905)IUSER,IBREAK,IUSER,OBJLIN,DELX
 0091             905 FORMAT(4X,'U0',I2,'X0',I1,3X,'USER0',I2,T25,F9.4,T40,'CONSOLE',
                     1T50,F9.4 )
 0092                 IF(IUSER .LT. 10) WRITE(NNUNIT,910)IUSER,IBREAK
 0093             910 FORMAT(' UP B',T15,'U00',I1,'X0',I1,T25,'1.0')
 0094                 IF(IUSER .GE. 10) WRITE(NNUNIT,911)IUSER,IBREAK
 0095             911 FORMAT(' UP B',T15,'U0',I2,'X0',I1,T25,'1.0')
 0096              21 CONTINUE
 0097                 IF(BREAK(IBREAK + 1) .GE. 20) GO TO 20
 0098                 IBREAK = IBREAK + 1
 0099                 GO TO 19
 0100              20 CONTINUE
 0101                 IF(IUSER.NE. NUMUSE) GO TO 22
```

```
FORTRAN IV G LEVEL 1, MOD 3                MAIN                DATE = 69034          23/16/11

 0102                    DO 22 NNUNIT = 6,7
                C        IF(NNUNIT .EQ. 7) GO TO 22
 0103                    DO 22 NN = 1,NUMUSE
 0104                    IF(NN .LT. 10) WRITE(NNUNIT,VF1) NN, RHSPI(NN)
 0105                    IF(NN .GE. 10) WRITE(NNUNIT,VF2) NN, RHSPI(NN)
 0106                 22 CONTINUE
 0107                  4 CONTINUE
 0108                    CALL EXIT
 0109                    END
```

```
FORTRAN IV G LEVEL 1, MOD 3            RHSF              DATE = 69034          23/16/11

   0001                    FUNCTION RHSF(PASSK,PI,NTESTO)
                 C         THIS FUNCTION COMPUTES THE LOWER BOUND ON X FOR THE CHANCE CONSTRAINT
   0002                    FLN = NTESTO
   0003                    PI = -1.0*PI
   0004                    TERM1 = 2.0*PASSK*FLN + (PI**2)
   0005                    TERM2 = (((PI**2)+4.0*FLN*PASSK*(1.-PASSK))**0.5)*PI
   0006                    TERM3 = 2.0*(FLN + (PI**2))
   0007                    RHSF = (TERM1 + TERM2) / (TERM3)
   0008                    RETURN
   0009                    END
```

```
FORTRAN IV G LEVEL 1, MOD 3              F                    DATE = 69034          23/16/11

     0001                 FUNCTION F(PHI,FKONE,FKTWO,T)
     0002                 F = 1./ (1.0 + (1. /EXP(FKTWO + FKONE *PHI*T)))
     0003                 RETURN
     0004                 END
```

```
17
18X,6HUNITS,T14,11HSERROR,I1,T29,F8,38
14X,6HUNITS,T15,11HSERCS,I2,T25,F8,38
[illegible]
.002260 -1.2960
[illegible]
.0046   -1.7170
.0081   -1.480
.0142   -3.1435
[illegible]
.0092   -2.100
[illegible]
.0132   -3.0816
.0033   -.5366
.0196   -3.2612
[illegible]
.0029   -2.0041
[illegible]
.0143   -3.3408
.0968   -3.8478
  .500  40.9
/*
```

37

```
          U001X01   LINOBJ        0.520     PHI001      0.100
          U001X01   USER001       0.5200    CONSOLE     0.1000
     UP B           U001X01   1.0
          U001X02   LINOBJ        0.162     PHI001      0.050
          U001X02   USER001       0.1621    CONSOLE     0.0500
     UP B           U001X02   1.0
          U001X03   LINOBJ        0.099     PHI001      0.050
          U001X03   USER001       0.0986    CONSOLE     0.0500
     UP B           U001X03   1.0
          U001X04   LINOBJ        0.051     PHI001      0.050
          U001X04   USER001       0.0506    CONSOLE     0.0500
     UP B           U001X04   1.0
          U001X05   LINOBJ        0.024     PHI001      0.050
          U001X05   USER001       0.0237    CONSOLE     0.0500
     UP B           U001X05   1.0
          U001X06   LINOBJ        0.015     PHI001      0.100
          U001X06   USER001       0.0154    CONSOLE     0.1000
     UP B           U001X06   1.0
          U001X07   LINOBJ        0.004     PHI001      0.550
          U001X07   USER001       0.0036    CONSOLE     0.5500
     UP B           U001X07   1.0
          U002X01   LINOBJ        0.569     PHI002      0.100
          U002X01   USER002       0.5687    CONSOLE     0.1000
     UP B           U002X01   1.0
          U002X02   LINOBJ        0.128     PHI002      0.050
          U002X02   USER002       0.1281    CONSOLE     0.0500
     UP B           U002X02   1.0
          U002X03   LINOBJ        0.075     PHI002      0.050
          U002X03   USER002     . 0.0753    CONSOLE     0.0500
     UP B           U002X03   1.0
          U002X04   LINOBJ        0.039     PHI002      0.050
          U002X04   USER002       0.0392    CONSOLE     0.0500
     UP B           U002X04   1.0
          U002X05   LINOBJ        0.019     PHI002      0.050
          U002X05   USER002       0.0191    CONSOLE     0.0500
     UP B           U002X05   1.0
          U002X06   LINOBJ        0.013     PHI002      0.100
          U002X06   USER002       0.0132    CONSOLE     0.1000
     UP B           U002X06   1.0
          U002X07   LINOBJ        0.004     PHI002      0.550
          U002X07   USER002       0.0036    CONSOLE     0.5500
     UP B           U002X07   1.0
          U003X01   LINOBJ        0.529     PHI003      0.100
          U003X01   USER003       0.5294    CONSOLE     0.1000
     UP B           U003X01   1.0
          U003X02   LINOBJ        0.126     PHI003      0.050
          U003X02   USER003       0.1261    CONSOLE     0.0500
     UP B           U003X02   1.0
          U003X03   LINOBJ        0.089     PHI003      0.050
          U003X03   USER003       0.0887    CONSOLE     0.0500
     UP B           U003X03   1.0
          U003X04   LINOBJ        0.056     PHI003      0.050
          U003X04   USER003       0.0557    CONSOLE     0.0500
     UP B           U003X04   1.0
          U003X05   LINOBJ        0.051     PHI003      0.100
          U003X05   USER003       0.0507    CONSOLE     0.1000
     UP B           U003X05   1.0
          U003X06   LINOBJ        0.015     PHI003      0.100
          U003X06   USER003       0.0153    CONSOLE     0.1000
     UP B           U003X06   1.0
          U003X07   LINOBJ        0.006     PHI003      0.500
          U003X07   USER003       0.0061    CONSOLE     0.5000
```

```
UP B           U014X05    1.0
    U014X06    LINOBJ        0.055       PHI014        0.050
    U014X06    USER014       0.0549      CONSOLE       0.0500
UP B           U014X06    1.0
    U014X07    LINOBJ        0.026       PHI014        0.050
    U014X07    USER014       0.0256      CONSOLE       0.0500
UP B           U014X07    1.0
    U014X08    LINOBJ        0.016       PHI014        0.100
    U014X08    USER014       0.0163      CONSOLE       0.1000
UP B           U014X08    1.0
    U014X09    LINOBJ        0.004       PHI014        0.500
    U014X09    USER014       0.0037      CONSOLE       0.5000
UP B           U014X09    1.0
    U015X01    LINOBJ        0.165       PHI015        0.050
    U015X01    USER015       0.1648      CONSOLF       0.0500
UP B           U015X01    1.0
    U015X02    LINOBJ        0.433       PHI015        0.100
    U015X02    USER015       0.4328      CONSOLE       0.1000
UP B           U015X02    1.0
    U015X03    LINOBJ        0.175       PHI015        0.050
    U015X03    USER015       0.1746      CONSOLE       0.0500
UP B           U015X03    1.0
    U015X04    LINOBJ        0.096       PHI015        0.050
    U015X04    USER015       0.0957      CONSOLE       0.0500
UP B           U015X04    1.0
    U015X05    LINOBJ        0.043       PHI015        0.050
    U015X05    USER015       0.0434      CONSOLE       0.0500
UP B           U015X05    1.0
    U015X06    LINOBJ        0.025       PHI015        0.100
    U015X06    USER015       0.0251      CONSOLE       0.1000
UP B           U015X06    1.0
    U015X07    LINOBJ        0.005       PHI015        0.550
    U015X07    USER015       0.0046      CONSOLE       0.5500
UP B           U015X07    1.0
    U016X01    LINOBJ        0.454       PHI016        0.050
    U016X01    USER016       0.4541      CONSOLE       0.0500
UP B           U016X01    1.0
    U016X02    LINOBJ        0.231       PHI016        0.050
    U016X02    USER016       0.2308      CONSOLE       0.0500
UP B           U016X02    1.0
    U016X03    LINOBJ        0.072       PHI016        0.050
    U016X03    USER016       0.0719      CONSOLE       0.0500
UP B           U016X03    1.0
    U016X04    LINOBJ        0.021       PHI016        0.100
    U016X04    USER016       0.0207      CONSOLE       0.1000
UP B           U016X04    1.0
    U016X05    LINOBJ        0.001       PHI016        0.700
    U016X05    USER016       0.0010      CONSOLE       0.7000
UP B           U016X05    1.0
    U017X01    LINOBJ        0.477       PHI017        0.050
    U017X01    USER017       0.4770      CONSOLE       0.0500
UP B           U017X01    1.0
    U017X02    LINOBJ        0.017       PHI017        0.050
    U017X02    USER017       0.0166      CONSOLE       0.0500
UP B           U017X02    1.0
    U017X03    LINOBJ        0.000       PHI017        0.850
    U017X03    USER017       0.0003      CONSOLE       0.8500
UP B           U017X03    1.0
    RHS        USER001       0.500
    RHS        USER002       0.500
    RHS        USER003       0.500
    RHS        USER004       0.500
    RHS        USER005       0.500
```

APPENDIX 7.2

CHAPTER SEVEN

MATHEMATICAL PROGRAMMING SYSTEM: CONTROL PROGRAM

Once the preceding program has prepared the data, a control program is needed to direct the solution of the problem. This control program sets up the data into a computable problem, directs the problem solution, directs various analyses of the solution and controls the general flow of operation of the Mathematical Programming System.

A listing of such a control program is reproduced at the end of this appendix. The program and its data will be used in the case presented in the next appendix. The lines from PROGRAM to PEND are commands to the MPS/360 in a special language.

The name of the first job is 'GTOWN SPRING LOGISTIC CHANCECON OBJECT'. The command INITIALZ, labeled A, establishes the frequency of reports and actions to be taken in the event of errors. The commands from A to B are all concerned with preliminaries to solving the scheduling problem. The two MOVE commands establish the names of the problem (XPBNAME) as 'SCHEDULE' and the data file (XDATA) as 'SCHEDULE'. The CONVERT card causes MPS/360 to convert the information coded onto punched cards into a properly formated problem on the problem file. A report that is generated by this conversion will be printed and given the heading 'SUMMARY'.

The next four cards down to but not including B are concerned with demands that may arise during the course of computation. XDELTM = 5 and MVADR (XDODELTM)--- cause the system to inspect calculations periodically and in the event that calculations proceed for 5 minutes without solution the results up to that point are stored.

The command MVADR (XDONFS, INFEAS) will transfer to the card labeled INFEAS, at the bottom of the program listing if infeasibility occurs during a solution sequence. If the problem 'SCHEDULE' has no feasible solution the system goes to INFEAS and executes TRACE which will print a report listing the extent of infeasibility, all vectors in the infeasible rows, and all infeasible rows. The command SOLUTION produces a complete output report, STATUS produces a detailed report on the status of the MPS/360 system, and the final command in the infeasibility sequence, EXIT, causes the system to abandon further calculation.

The third MVADR card, MVADR (XDOFEAS, SAVES) causes the first feasible solution that arises during a solution sequence to be saved under the name 'SCHED'. This will be done by transfering to the card

SAVES when a feasible solution is found. At this point the preliminaries have been completed. Various provisions for pathological situations have been made and the system can be turned to solving the scheduling problem represented by the input data deck.

The card marked B, SETUP, causes MPS/360 to use facts gathered during the convert phase to allocate memory for storing the intermediate results of calculation. A large amount of memory is required, extensive use is made of disc storage. This card also informs MPS/360 that the problem is to be set up for minimization with the variable bounded by the vector 'B'.

The objective is defined as 'FULLOBJ' and the right hand side vector is defined as 'RHS'. The problem is now completely specified. Unless there is some error in the data, the computer is ready to carry out a solution to the problem named 'SCHEDULE'. Since we are using the simplex algorithm there is a choice between methods of solution. We may solve either the primal linear programming problem or its equivalent dual. In our case there are several times as many columns as there are rows, so solution of the primal is called for. The command PRIMAL causes MPS/360 to use the simplex algorithm (with modifications for the separable programming formulation) to solve the primal linear programming problem. SOLUTION causes the optimal solution to be reported. The final solution, once it is found, is stored under the name 'OPTSCHED' for use later in parametric studies.

The next portion of the control program from TESTRHS to TESTOBJ defines a new problem 'PARA-R-H-S'. This problem restores the optimal solution to 'SCHEDULE' and prepares for a sensitivity study of the optimal solution 'OPTSCHED' as the elements of the RHS vector are changed.

The right hand side will be varied by adding a constant times the change column RH2. The values of this constant range from XPARAM = 0 to XPARAM = 10. In increments of XPARDELT the system prints a report on the course of the parametric study.

The final portion of the control program is similar to the last except that the sensitivity of the optimal solution to changes in the objective will be studied.

DATA DECK

There are some data cards not prepared by the data preparation program of Appendix 7. 1. Since the listing at the end of this appendix is complete, these cards will be described briefly here.

The first card NAME is self explanatory, it defines this data file as 'SCHEDULE'. The next card ROWS signals that the subsequent cards are the names of the rows in the data matrix. In the listing shown the rows PHIXXX are dummy rows introduced to keep track of the time allocations to each user. The N preceding each name indicates that this row does not enter into the constraint set. The row LINOBJ is the actual objective function of the problem. FULLOBJ is set equal to LINOBJ and FULLOBJ is used as the final objective in the problem. This arrangement of the objective function is not required for the chance constrained formulation but was used to solve other formulations and has been kept.

The rows labeled USERXXX establish the constraints on passing grades for each student. The G before each row indicates a greater than (≥) inequality sign for that row. The final row is CONSOLE marked with an L for less than (≤). This row is the capacity constraint.

The next card, COLUMNS, signals that all of the rows of the problem have been named and that the next data will define the columns and the coefficients that will be put into the data matrix. The 'SEPORG' 'MARKXX' card and the 'SEPEND' 'MARK5XX' cards are devices to define sets of restricted variables for separable programming. All of the columns corresponding to one student's learning curve are set-off by two of these cards.

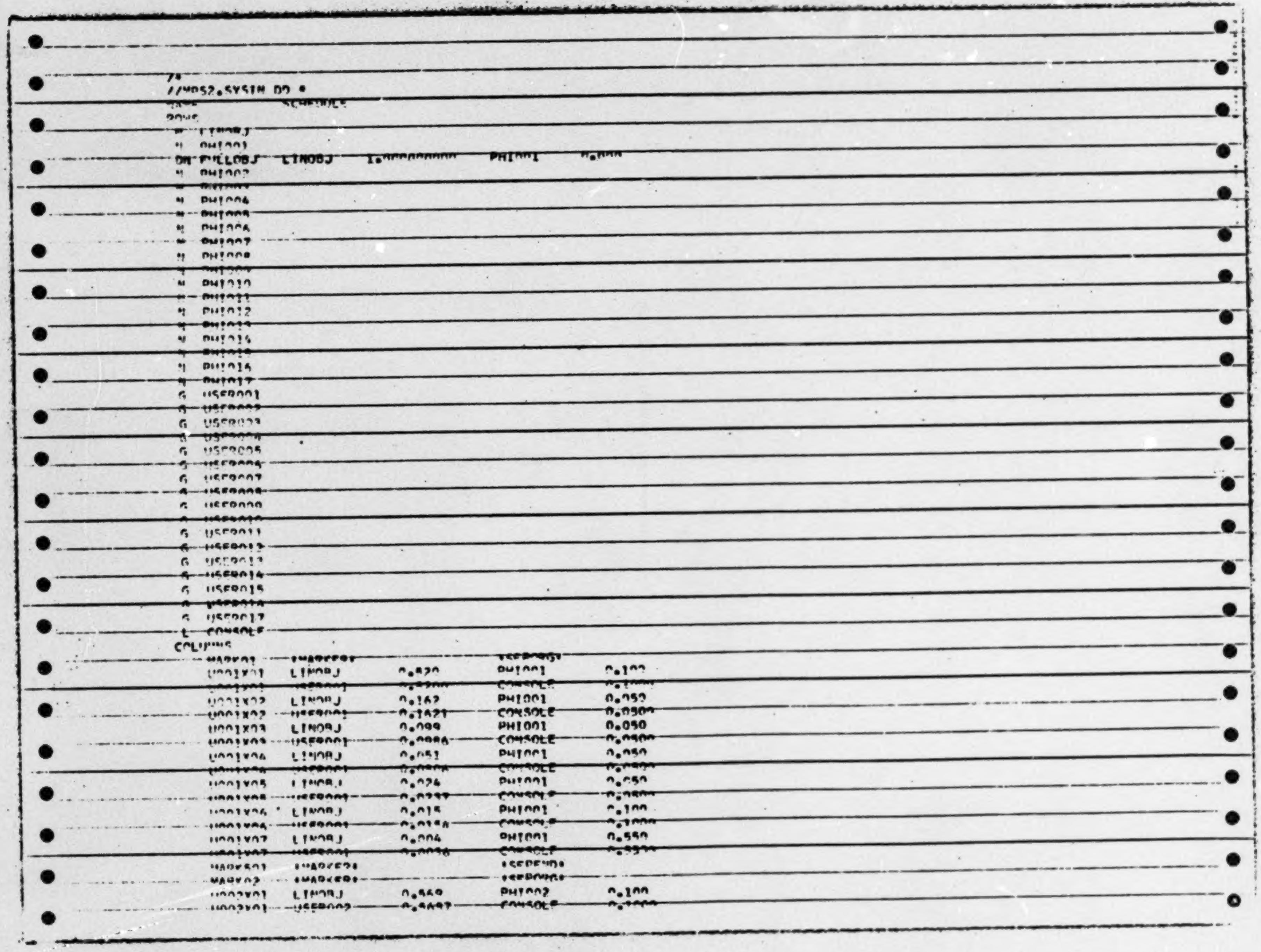
The first card in the columns section indicates that column U001X01 and row LINOBJ has the value 0.520. For each column the coefficients are indicated by row, so there are values for LINOBJ, PHIXXX--the dummy row, USERXXX--the passing grade constraint, and CONSOLE--the capacity constraint. In this way every column of each user's submatrix has its non zero elements specified. All of the cards required for this are produced by the earlier data preparation program. The restricted variables for each user are demarcated by the marker cards.

The completion of the columns section is signalled by the RHS card. This indicates that the right hand side is to be defined. In the data deck listed we have defined two right hand side vectors, RHS and RHZ; the first is to be used in problem SCHEDULE, while the second will be used to study the effect of reducing the available console capacity.

The last section of the data deck consists of the BOUNDS section. This section provides the upper limits on the value of any variable. Because of the formulation of this problem, all variables have an upper bound of 1.0.

On the following pages is a listing of the MPS/360 control program and data deck.

```
     CONTROL PROGRAM COMPILER

0001                PROGRAM
[illegible]         TITLE('[illegible] SPRING LOGISTIC CHANCECON OBJECT')
0003      A         INITIALZ
[illegible]         MOVE([illegible]'SCHEDULE')
0063                MOVE(XPRNAME,'SCHEDULE')
[illegible]         CONVERT('S[illegible]')
0065                VDELTA=5
[illegible]         MVADD(XDDELTA,SAVES)
0067                MVADD(XDONES,INFEAS)
[illegible]         [illegible]
0069      B         SETUP('MINIMUM','M','MAX')
[illegible]         MOVE(XOBJ,'FULLOBJ')
0071                MOVE(XRHS,'RHS')
[illegible]         PRIMAL
0073                SOLUTION
[illegible]         SAVE([illegible])
0075      TESTRHS   TITLE('TEST R H S')
[illegible]         RESTORE('NAME','OPTSCHED')
0077                XPARAM=0.
[illegible]         XPARMAX=10.
0079                XPARDELT=10.
[illegible]         MOVE(XCHECK,'RHS2')
0081                PARARHS
[illegible]         SOLUTION
0083      TESTOBJ   TITLE('PARAOBJ')
[illegible]         RESTORE('NAME','OPTSCHED')
0085                XPARAM=0.
[illegible]         XPARMAX=10.
0087                XPARDELT=10.
[illegible]         MOVE([illegible])
0089                MOVE(XOBJ,'FULLOBJ')
[illegible]         PARAOBJ
0091                SOLUTION
[illegible]         EXIT
0093      SAVES     SAVE('NAME','SCHED')
[illegible]         CONTINUE
0095      INFEAS    TRACE
[illegible]         SOLUTION
0097                STATUS
[illegible]         EXIT
0099                PEND
```



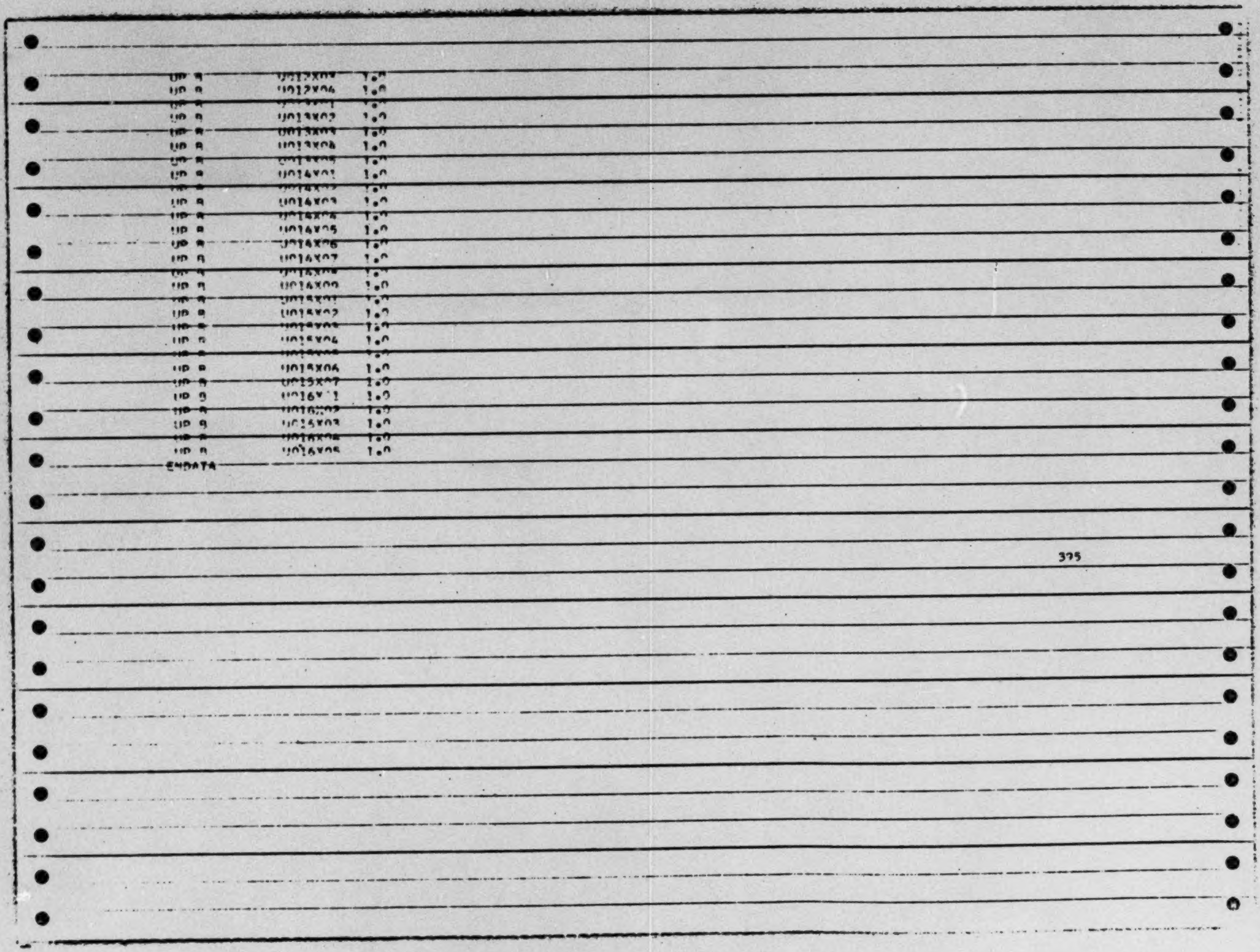
```
/*
//MPS2.SYSIN DD *
NAME          SCHEDULE
ROWS
 N  LINOBJ
 N  PHI001
 DN FULLOBJ   LINOBJ      1.000000000    PHI001      0.000
 N  PHI002
 N  PHI003
 N  PHI004
 N  PHI005
 N  PHI006
 N  PHI007
 N  PHI008
 N  PHI009
 N  PHI010
 N  PHI011
 N  PHI012
 N  PHI013
 N  PHI014
 N  PHI015
 N  PHI016
 N  PHI017
 G  USER001
 G  USER002
 G  USER003
 G  USER004
 G  USER005
 G  USER006
 G  USER007
 G  USER008
 G  USER009
 G  USER010
 G  USER011
 G  USER012
 G  USER013
 G  USER014
 G  USER015
 G  USER016
 G  USER017
 L  CONSOLE
COLUMNS
    MARK01    'MARKER'                 'USERORG'
    U001X01   LINOBJ        0.820      PHI001        0.100
    U001X01   USER001       0.[illegible]      CONSOLE       0.1000
    U001X02   LINOBJ        0.162      PHI001        0.050
    U001X02   USER001       0.1421     CONSOLE       0.0500
    U001X03   LINOBJ        0.099      PHI001        0.050
    U001X03   USER001       0.0986     CONSOLE       0.0500
    U001X04   LINOBJ        0.051      PHI001        0.050
    U001X04   USER001       0.[illegible]      CONSOLE       0.0500
    U001X05   LINOBJ        0.024      PHI001        0.050
    U001X05   USER001       0.0237     CONSOLE       0.0500
    U001X06   LINOBJ        0.015      PHI001        0.100
    U001X06   USER001       0.0148     CONSOLE       0.1000
    U001X07   LINOBJ        0.004      PHI001        0.550
    U001X07   USER001       0.0036     CONSOLE       0.5500
    MARKE01   'MARKER'                 'USEREND'
    MARK02    'MARKER'                 'USERORG'
    U002X01   LINOBJ        0.560      PHI002        0.100
    U002X01   USER002       0.5497     CONSOLE       0.1000
```

```
U002X02   LINOBJ     0.125       PHI002      0.050
U002X02   USER002    0.1291      CONSOLE     0.0500
U002X03   LINOBJ     [illegible] PHI002      [illegible]
U002X03   USER002    0.0753      CONSOLE     [illegible]
U002X04   LINOBJ     0.039       PHI002      0.150
U002X04   USER002    0.0392      CONSOLE     0.0500
U002X05   LINOBJ     0.019       PHI002      0.050
U002X05   USER002    0.0191      CONSOLE     0.0500
U002X06   LINOBJ     0.013       PHI002      0.100
U002X06   USER002    0.0133      CONSOLE     0.1000
U002X07   LINOBJ     0.004       PHI002      0.550
U002X07   USER002    0.0036      CONSOLE     0.5500
MARKER02  'MARKER'               'USEREND'
MARK03    'MARKER'               'USERORG'
U003X01   LINOBJ     0.530       PHI003      0.100
U003X01   USER003    0.5304      CONSOLE     0.1000
U003X02   LINOBJ     0.126       PHI003      0.050
U003X02   USER003    0.1261      CONSOLE     0.0500
U003X03   LINOBJ     0.089       PHI003      0.050
U003X03   USER003    0.0887      CONSOLE     0.0500
U003X04   LINOBJ     0.056       PHI003      0.050
U003X04   USER003    0.0557      CONSOLE     0.0500
U003X05   LINOBJ     0.051       PHI003      0.100
U003X05   USER003    0.0507      CONSOLE     0.1000
U003X06   LINOBJ     0.015       PHI003      0.100
U003X06   USER003    0.0153      CONSOLE     0.1000
U003X07   LINOBJ     0.006       PHI003      0.500
U003X07   USER003    0.0061      CONSOLE     0.5000
MARKER03  'MARKER'               'USEREND'
MARK04    'MARKER'               'USERORG'
U004X01   LINOBJ     0.250       PHI004      0.050
U004X01   USER004    0.2504      CONSOLE     0.0500
U004X02   LINOBJ     [illegible] PHI004      0.150
U004X02   USER004    0.3547      CONSOLE     0.1500
U004X03   LINOBJ     0.097       PHI004      0.050
U004X03   USER004    0.0969      CONSOLE     0.0500
U004X04   LINOBJ     0.126       PHI004      0.100
U004X04   USER004    0.1264      CONSOLE     0.1000
U004X05   LINOBJ     [illegible] PHI004      0.100
U004X05   USER004    0.0582      CONSOLE     0.1000
U004X06   LINOBJ     0.024       PHI004      0.100
U004X06   USER004    0.0236      CONSOLE     0.1000
U004X07   LINOBJ     0.014       PHI004      [illegible]
U004X07   USER004    0.0142      CONSOLE     [illegible]
MARKER04  'MARKER'               'USEREND'
MARK05    'MARKER'               'USERORG'
U005X01   LINOBJ     0.575       PHI005      0.150
U005X01   USER005    0.5757      CONSOLE     0.1500
U005X02   LINOBJ     0.128       PHI005      0.050
U005X02   USER005    0.1277      CONSOLE     0.0500
U005X03   LINOBJ     [illegible] PHI005      [illegible]
U005X03   USER005    0.1333      CONSOLE     0.1000
U005X04   LINOBJ     [illegible] PHI005      [illegible]
U005X04   USER005    0.0663      CONSOLE     0.1000
U005X05   LINOBJ     [illegible] PHI005      [illegible]
U005X05   USER005    0.0300      CONSOLE     0.1000
U005X06   LINOBJ     [illegible] PHI005      [illegible]
U005X06   USER005    0.0103      CONSOLE     0.4500
MARKER05  'MARKER'               'USEREND'
MARK06    'MARKER'               'USERORG'
U006X01   LINOBJ     [illegible] PHI006      [illegible]
```

```
    U006X01   USER006    0.6846    CONSOLE    0.1000
    U006X02   LINOBJ     0.143     PHI006     0.050
    U006X02   USER006    0.1430    CONSOLE    0.0500
    U006X03   LINOBJ     0.038     PHI006     0.050
    U006X03   USER006    0.0380    CONSOLE    0.0500
    U006X04   LINOBJ     0.009     PHI006     0.050
    U006X04   USER006    0.0087    CONSOLE    0.0500
    U006X05   LINOBJ     0.002     PHI006     0.700
    U006X05   USER006    0.0024    CONSOLE    0.7000
    MARKS06   'MARKER'             'SEPEND'
    MARKS07   'MARKER'             'SEPORG'
    U007X01   LINOBJ     0.421     PHI007     0.050
    U007X01   USER007    0.4213    CONSOLE    0.0500
    U007X02   LINOBJ     0.270     PHI007     0.050
    U007X02   USER007    0.2706    CONSOLE    0.0500
    U007X03   LINOBJ     0.045     PHI007     0.150
    U007X03   USER007    0.0456    CONSOLE    0.1500
    U007X04   LINOBJ     0.029     PHI007     0.100
    U007X04   USER007    0.0293    CONSOLE    0.1000
    U007X05   LINOBJ     0.002     PHI007     0.700
    U007X05   USER007    0.0015    CONSOLE    0.7000
    MARKS07   'MARKER'             'SEPEND'
    MARKS08   'MARKER'             'SEPORG'
    U008X01   LINOBJ     0.565     PHI008     0.100
    U008X01   USER008    0.5654    CONSOLE    0.1000
    U008X02   LINOBJ     0.161     PHI008     0.050
    U008X02   USER008    0.1606    CONSOLE    0.0500
    U008X03   LINOBJ     0.080     PHI008     0.050
    U008X03   USER008    0.0803    CONSOLE    0.0500
    U008X04   LINOBJ     0.034     PHI008     0.050
    U008X04   USER008    0.0338    CONSOLE    0.0500
    U008X05   LINOBJ     0.018     PHI008     0.100
    U008X05   USER008    0.0182    CONSOLE    0.1000
    U008X06   LINOBJ     0.003     PHI008     0.600
    U008X06   USER008    0.0030    CONSOLE    0.6000
    MARKS08   'MARKER'             'SEPEND'
    MARKS09   'MARKER'             'SEPORG'
    U009X01   LINOBJ     0.444     PHI009     0.050
    U009X01   USER009    0.4445    CONSOLE    0.0500
    U009X02   LINOBJ     0.181     PHI009     0.050
    U009X02   USER009    0.1813    CONSOLE    0.0500
    U009X03   LINOBJ     0.078     PHI009     0.050
    U009X03   USER009    0.0777    CONSOLE    0.0500
    U009X04   LINOBJ     0.027     PHI009     0.050
    U009X04   USER009    0.0277    CONSOLE    0.0500
    U009X05   LINOBJ     0.012     PHI009     0.100
    U009X05   USER009    0.0117    CONSOLE    0.1000
    U009X06   LINOBJ     0.001     PHI009     0.650
    U009X06   USER009    0.0014    CONSOLE    0.6500
    MARKS09   'MARKER'             'SEPEND'
    MARKS10   'MARKER'             'SEPORG'
    U010X01   LINOBJ     0.224     PHI010     0.050
    U010X01   USER010    0.2238    CONSOLE    0.0500
    U010X02   LINOBJ     0.325     PHI010     0.050
    U010X02   USER010    0.3247    CONSOLE    0.0500
    U010X03   LINOBJ     0.166     PHI010     0.050
    U010X03   USER010    0.1666    CONSOLE    0.0500
    U010X04   LINOBJ     0.052     PHI010     0.050
    U010X04   USER010    0.0517    CONSOLE    0.0500
    U010X05   LINOBJ     0.017     PHI010     0.100
    U010X05   USER010    0.0167    CONSOLE    0.1000
```

```
U010X06   LIMORJ     0.001     PHI010     0.650
U010X06   USER010    0.0011    CONSOLE    0.6500
MARKE10   'MARKER'             'SEPEND'
MARK11    'MARKER'             'SEPORG'
U011X01   LIMORJ     0.506     PHI011     0.100
U011X01   USER011    0.5059    CONSOLE    0.1000
U011X02   LIMORJ     0.144     PHI011     0.100
U011X02   USER011    0.1443    CONSOLE    0.1000
U011X03   LIMORJ     0.109     PHI011     0.100
U011X03   USER011    0.1086    CONSOLE    0.1000
U011X04   LIMORJ     0.071     PHI011     0.100
U011X04   USER011    0.0711    CONSOLE    0.1000
U011X05   LIMORJ     0.062     PHI011     0.100
U011X05   USER011    0.0624    CONSOLE    0.1000
U011X06   LIMORJ     0.024     PHI011     0.100
U011X06   USER011    0.0230    CONSOLE    0.1000
U011X07   LIMORJ     0.017     PHI011     0.150
U011X07   USER011    0.0170    CONSOLE    0.1500
U011X08   LIMORJ     0.008     PHI011     0.200
U011X08   USER011    0.0078    CONSOLE    0.2000
MARKE11   'MARKER'             'SEPEND'
MARK12    'MARKER'             'SEPORG'
U012X01   LIMORJ     0.484     PHI012     0.050
U012X01   USER012    0.4838    CONSOLE    0.0500
U012X02   LIMORJ     0.279     PHI012     0.050
U012X02   USER012    0.2790    CONSOLE    0.0500
U012X03   LIMORJ     0.057     PHI012     0.[illegible]
U012X03   USER012    0.0573    CONSOLE    0.[illegible]
U012X04   LIMORJ     0.010     PHI012     0.[illegible]
U012X04   USER012    0.0096    CONSOLE    0.8000
MARKE12   'MARKER'             'SEPEND'
MARK13    'MARKER'             'SEPORG'
U013X01   LIMORJ     0.683     PHI013     0.100
U013X01   USER013    0.6828    CONSOLE    0.1000
U013X02   LIMORJ     0.148     PHI013     0.050
U013X02   USER013    0.1481    CONSOLE    0.0500
U013X03   LIMORJ     0.030     PHI013     0.050
U013X03   USER013    0.0303    CONSOLE    0.0500
U013X04   LIMORJ     0.009     PHI013     0.050
U013X04   USER013    0.0089    CONSOLE    0.0500
U013X05   LIMORJ     0.002     PHI013     0.700
U013X05   USER013    0.0025    CONSOLE    0.7000
MARKE13   'MARKER'             'SEPEND'
MARK14    'MARKER'             'SEPORG'
U014X01   LIMORJ     0.169     PHI014     0.050
U014X01   USER014    0.1689    CONSOLE    0.0500
U014X02   LIMORJ     0.181     PHI014     0.050
U014X02   USER014    0.1814    CONSOLE    0.0500
U014X03   LIMORJ     0.209     PHI014     0.050
U014X03   USER014    0.2089    CONSOLE    0.0500
U014X04   LIMORJ     0.172     PHI014     0.050
U014X04   USER014    0.1721    CONSOLE    0.0500
U014X05   LIMORJ     0.107     PHI014     0.050
U014X05   USER014    0.1066    CONSOLE    0.0500
U014X06   LIMORJ     0.055     PHI014     0.050
U014X06   USER014    0.0549    CONSOLE    0.0500
U014X07   LIMORJ     0.025     PHI014     0.050
U014X07   USER014    0.0254    CONSOLE    0.0500
U014X08   LIMORJ     0.016     PHI014     0.100
U014X08   USER014    0.0163    CONSOLE    0.1000
U014X09   LIMORJ     0.006     PHI014     0.800
```

```
    U014X09   USER014    0.0037     CONSOLE    0.5100
    MARKS14   'MARKER'              'SEPEND'
    MARK15    'MARKER'              'SEPORG'
    U015X01   LINOBJ     0.165      PHI015     0.050
    U015X01   USER015    0.1648     CONSOLE    0.0500
    U015X02   LINOBJ     0.433      PHI015     0.100
    U015X02   USER015    0.4328     CONSOLE    0.1000
    U015X03   LINOBJ     0.175      PHI015     0.050
    U015X03   USER015    0.1746     CONSOLE    0.0500
    U015X04   LINOBJ     0.096      PHI015     0.050
    U015X04   USER015    0.0957     CONSOLE    0.0500
    U015X05   LINOBJ     0.043      PHI015     0.050
    U015X05   USER015    0.0426     CONSOLE    0.0500
    U015X06   LINOBJ     0.025      PHI015     0.100
    U015X06   USER015    0.0251     CONSOLE    0.1000
    U015X07   LINOBJ     0.005      PHI015     0.550
    U015X07   USER015    0.0046     CONSOLE    0.5500
    MARKS15   'MARKER'              'SEPEND'
    MARK16    'MARKER'              'SEPORG'
    U016X01   LINOBJ     0.454      PHI016     0.050
    U016X01   USER016    [illegible] CONSOLE   0.0500
    U016X02   LINOBJ     0.231      PHI016     0.050
    U016X02   USER016    0.2308     CONSOLE    0.0500
    U016X03   LINOBJ     0.072      PHI016     0.050
    U016X03   USER016    0.0719     CONSOLE    0.0500
    U016X04   LINOBJ     0.021      PHI016     0.100
    U016X04   USER016    [illegible] CONSOLE   0.1000
    U016X05   LINOBJ     0.001      PHI016     0.700
    U016X05   USER016    0.0010     CONSOLE    0.7000
    MARKS16   'MARKER'              'SEPEND'
RHS
    RHS       USER001    0.500
    RHS       USER002    0.500
    RHS       USER003    0.500
    RHS       USER004    0.500
    RHS       USER005    0.500
    RHS       USER006    0.500
    RHS       USER007    0.500
    RHS       USER008    0.500
    RHS       USER009    0.500
    RHS       USER010    0.500
    RHS       USER011    0.500
    RHS       USER012    0.500
    RHS       USER013    0.500
    RHS       USER014    0.500
    RHS       USER015    0.500
    RHS       USER016    0.500
    RHS       CONSOLE   4.000
    RHS       CONSOLE   -0.25
BOUNDS
 UP B         U001X01   1.0
 UP B         U001X02   1.0
 UP B         U001X03   1.0
 UP B         U001X04   1.0
 UP B         U001X05   1.0
 UP B         U001X06   1.0
 UP B         U001X07   1.0
 UP B         U002X01   1.0
 UP B         U002X02   1.0
 UP B         U002X03   1.0
 UP B         U002X04   1.0
UP B      U002X05   1.0
UP B      U002X06   1.0
UP B      U002X07   1.0
UP B      U003X01   1.0
UP B      U003X02   1.0
UP B      U003X03   1.0
UP B      U003X04   1.0
UP B      U003X05   1.0
UP B      U003X06   1.0
UP B      U003X07   1.0
UP B      U004X01   1.0
UP B      U004X02   1.0
UP B      U004X03   1.0
UP B      U004X04   1.0
UP B      U004X05   1.0
UP B      U004X06   1.0
UP B      U004X07   1.0
UP B      U005X01   1.0
UP B      U005X02   1.0
UP B      U005X03   1.0
UP B      U005X04   1.0
UP B      U005X05   1.0
UP B      U005X06   1.0
UP B      U006X01   1.0
UP B      U006X02   1.0
UP B      U006X03   1.0
UP B      U006X04   1.0
UP B      U006X05   1.0
UP B      U007X01   1.0
UP B      U007X02   1.0
UP B      U007X03   1.0
UP B      U007X04   1.0
UP B      U007X05   1.0
UP B      U008X01   1.0
UP B      U008X02   1.0
UP B      U008X03   1.0
UP B      U008X04   1.0
UP B      U008X05   1.0
UP B      U008X06   1.0
UP B      U009X01   1.0
UP B      U009X02   1.0
UP B      U009X03   1.0
UP B      U009X04   1.0
UP B      U009X05   1.0
UP B      U009X06   1.0
UP B      U010X01   1.0
UP B      U010X02   1.0
UP B      U010X03   1.0
UP B      U010X04   1.0
UP B      U010X05   1.0
UP B      U010X06   1.0
UP B      U011X01   1.0
UP B      U011X02   1.0
UP B      U011X03   1.0
UP B      U011X04   1.0
UP B      U011X05   1.0
UP B      U011X06   1.0
UP B      U011X07   1.0
UP B      U011X08   1.0
UP B      U012X01   1.0
UP B      U012X02   1.0
```



```
 UP B          U012X05   1.0
 UP B          U012X06   1.0
 UP B          U013X01   1.0
 UP B          U013X02   1.0
 UP B          U013X03   1.0
 UP B          U013X04   1.0
 UP B          U013X05   1.0
 UP B          U014X01   1.0
 UP B          U014X02   1.0
 UP B          U014X03   1.0
 UP B          U014X04   1.0
 UP B          U014X05   1.0
 UP B          U014X06   1.0
 UP B          U014X07   1.0
 UP B          U014X08   1.0
 UP B          U014X09   1.0
 UP B          U015X01   1.0
 UP B          U015X02   1.0
 UP B          U015X03   1.0
 UP B          U015X04   1.0
 UP B          U015X05   1.0
 UP B          U015X06   1.0
 UP B          U015X07   1.0
 UP B          U016X01   1.0
 UP B          U016X02   1.0
 UP B          U016X03   1.0
 UP B          U016X04   1.0
 UP B          U016X05   1.0
ENDATA
```

375

APPENDIX 7.3

CHAPTER SEVEN

SOLUTION OF A SCHEDULING PROBLEM: A CASE

This appendix displays and interprets the results of using the MPS/360 spearable programming algorithm to solve a sample scheduling problem. The data and control program used in this case has already been discussed in Appendices 7.1 and 7.2.

The MPS/360 prints a running commentary on the progress of setting up and solving a particular problem. The reports presented here result when an optimal schedule is found. Five pages at the end of this appendix have been reproduced from the output of the programming system. The first is the optimal solution to scheduling the group of 16 students onto four consoles. Report 7.3.2 (covering three pages) traces the change in the optimal solution as the console capacity is varied from the original value of 4.0 to a new value of 1.5 consoles.

Optimal Solution

Report 7.3.1, at the back of this appendix, lists the optimal schedule for the 16 students in this case on 4.0 consoles. There are two rows for each user (XXX) they are PHIXXX and USERXXX.

The activity level in the PHIXXX row is the proportion of course time that is scheduled for student user XXX. The activity level in row USERXXX is that same user's expected final test score.

The value of the objective function FULLOBJ is found in row 3. The value 13.3365 is the sum of all students' predicted final test scores assuming that the schedule is implemented and that the students' learning behavior is accurately represented by the original parameter estimates for each student. This value corresponds to an average final achievement test score of 83.5%.

The only row which has a non-zero shadow price is the capacity constraint. It indicates that we might expect a marginal improvement of (0.55/16) in the average class achievement test score if we could increase the number of consoles by one unit. Of course, this is only a marginal result and a schedule consuming 5 consoles is not guaranteed to result in this improvement.

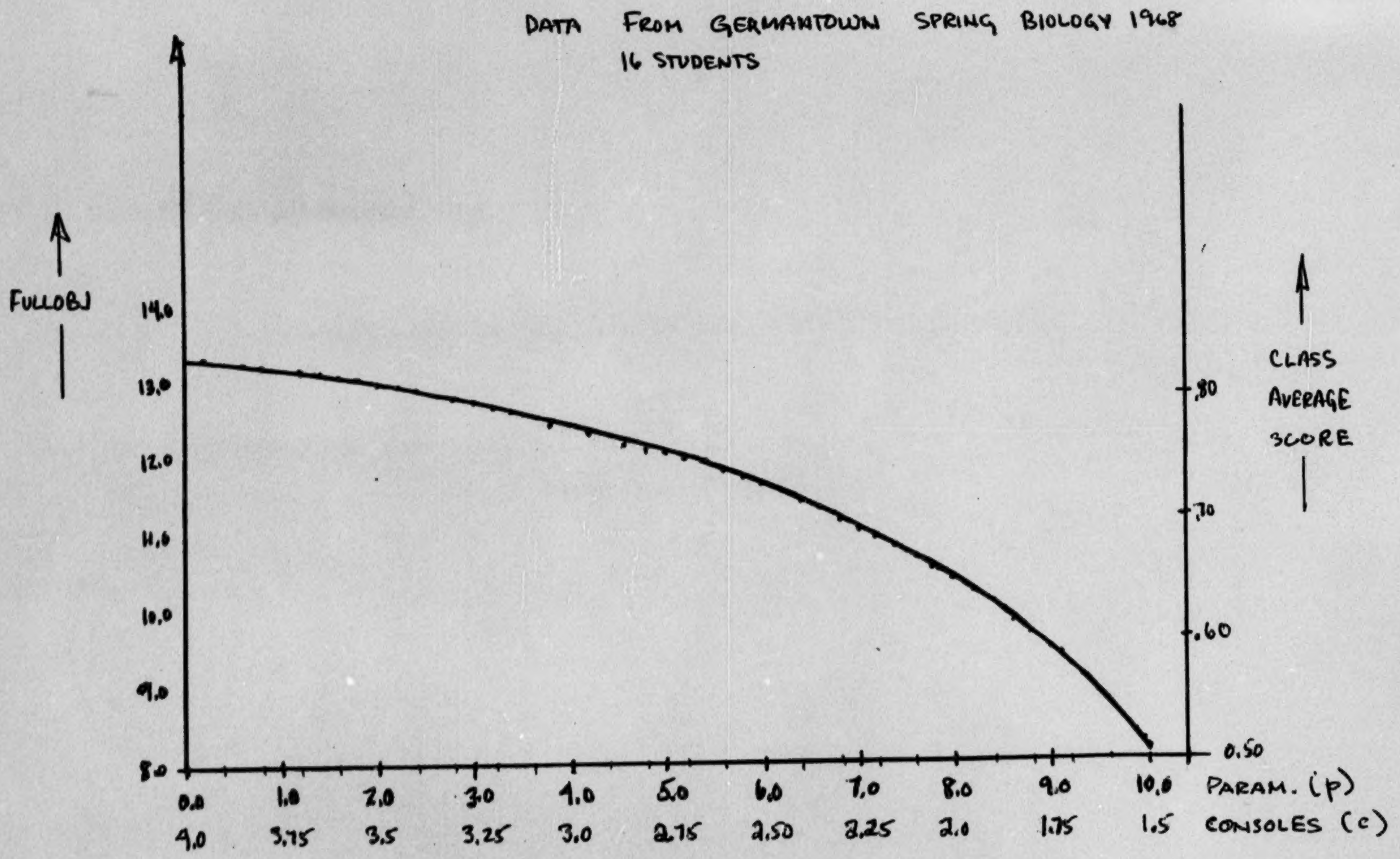
Sensitivity Analysis

Suppose that for some reason we actually have only 1.5 consoles and not four as in the preceding discussion. The parametric programming features of MPS/360 can be used to determine how this change in capacity will affect the optimal solution to the four console problems. Naturally, we would expect the average grade to fall if we have less console time available. The method used to perform this analysis is to add a quantity $p \times (-0.25)$ to the original console capacity of 4.0. p is a parameter used to obtain different ultimate values of the console capacity. In this case we generate all different optimal schedules for values of the console capacity (c) given by

$$c = 4.0 + p \times (-0.25) \qquad 0 \leq p \leq 10.0$$

When p achieves the value 10.0 we will have the schedule for c = 1.5 consoles, the case of interest to us. As an interesting aside, the MPS/360 computes the value of the objective function at each change of basic feasible solution caused by varying the right hand side of a constraint.

Report 7.3.2 (3 pages) details the reduction in the total class test score (FULLOBJ) as the console capacity is reduced by incremental increases of p. The column headed FUNCTION VALUE lists the value of the objective. The class average score as a function of the capacity available for the solution is plotted in Figure 7.3.1. This figure graphically presents the function value FULLOBJ plotted against PARAM (p) or its equivalent, console capacity. Class achievement does, in fact, decline smoothly as console capacity declines.



DATA FROM GERMANTOWN SPRING BIOLOGY 1968
16 STUDENTS

FIGURE 7.3.1
REDUCTION IN AVERAGE GRADE WITH DECREASING CONSOLE CAPACITY

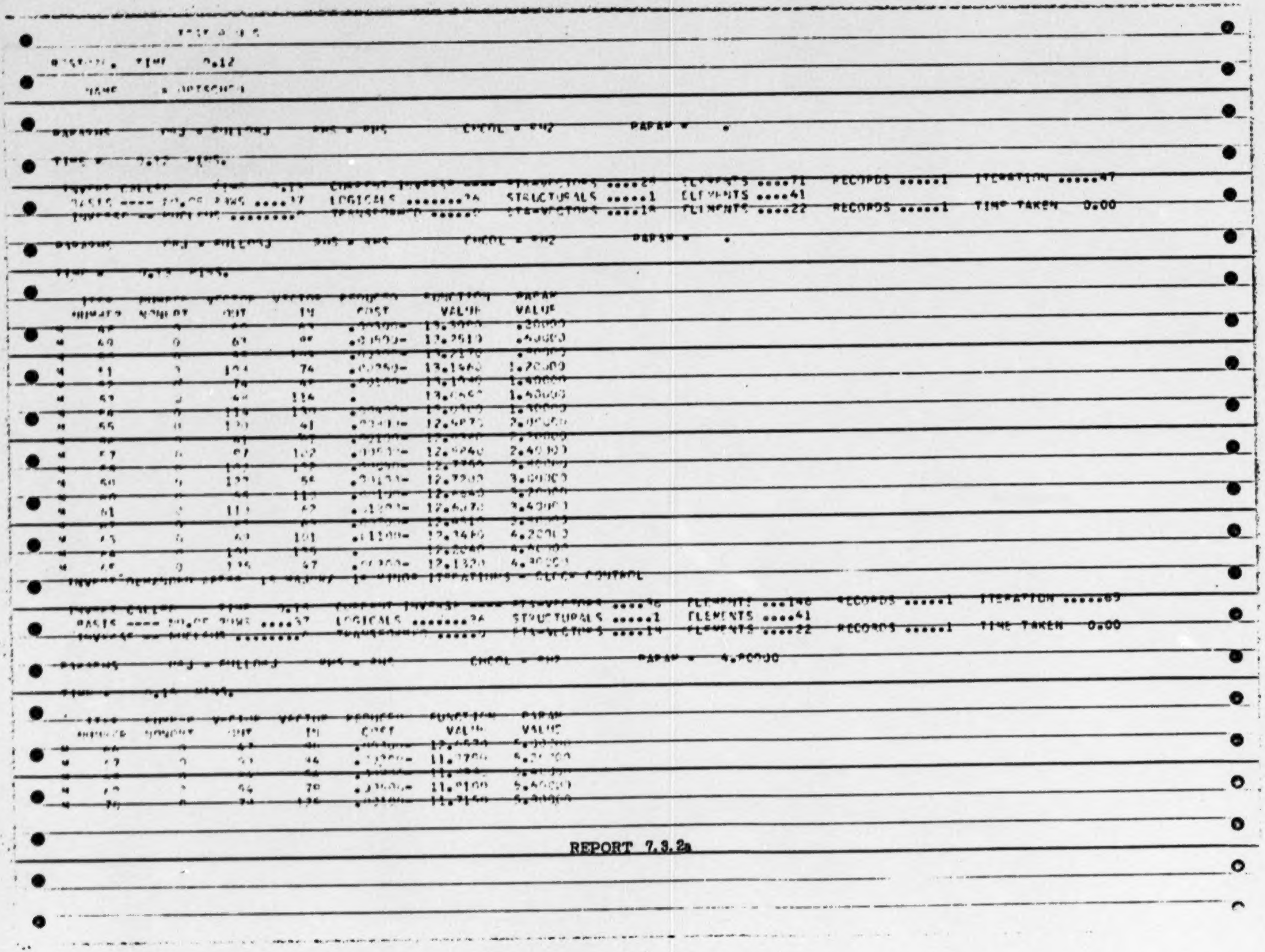
Report 7.3.3 is the summary of the optimal solution to the problem of scheduling this group of students onto 1.5 consoles. A look at the vector of shadow prices--DUAL ACTIVITY, shows that all students except USER006 are now constrained by their passing grade constraint. USER004 has a very high shadow price associated with his passing grade constraint. If it were necessary to drop any student from consideration for CAI, to drop student number 4 would have the largest impact on the achievement of all other students.

The shadow price on console capacity is -6.84, a much larger value than when there were four consoles (then it was -0.55 from Report 7.3.1). When we have only 1.5 consoles for this group of students an added console is a very valuable commodity in terms of the expected increase in average achievement for the whole class.

The approach that has been used in this case to study the effect of reducing console capacity may also be used when we are forced to reformulate an infeasible scheduling problem. We merely assume some large console capacity and then reduce the capacity parametrically until infeasibility occurs. The SOLUTION command in the infeasibility branch of the control program will cause the last solution (infeasible) to be printed. This report contains a wealth of information on the consequences of alternative reformulations.

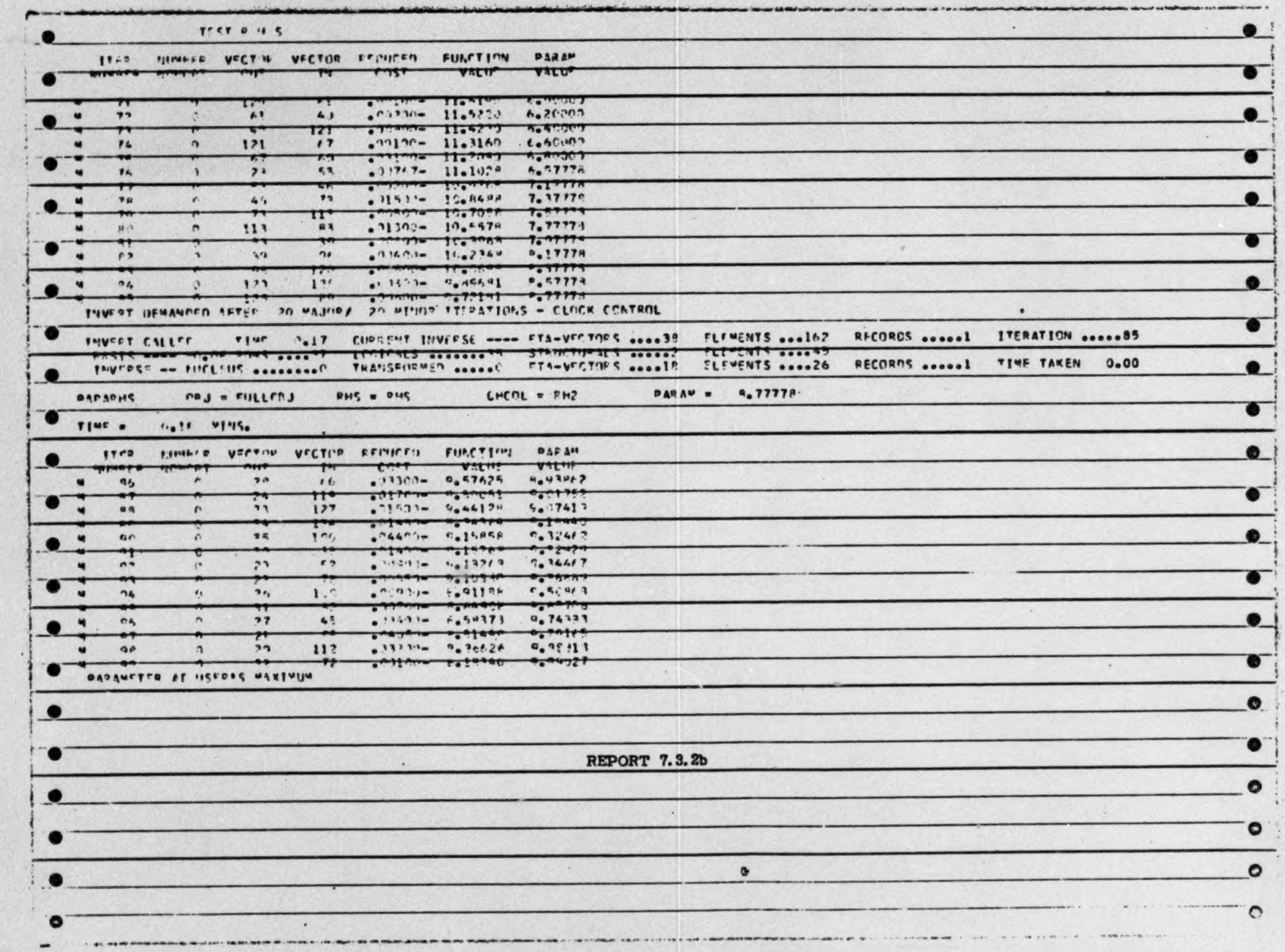
```
        [illegible] PROJECT

 SECTION 1 - [illegible]

 [illegible]   [illegible]   [illegible]   ...ACTIVITY...   SLACK ACTIVITY   ..LOWER LIMIT.   ..UPPER LIMIT.   .DUAL ACTIVITY

      1   [illegible]   BS    [illegible]    [illegible]     NONE       NONE        .
      2   [illegible]   BS      .25000         .25000-       NONE       NONE        .
      3   [illegible]   BS    [illegible]    [illegible]     NONE       NONE      1.00000
      4   [illegible]   BS      .25000         .25000-       NONE       NONE        .
      5   [illegible]   BS    [illegible]    [illegible]     NONE       NONE        .
      6   [illegible]   BS      .45000         .45000-       NONE       NONE        .
      7   [illegible]   BS    [illegible]      .35000-       NONE       NONE        .
      8   [illegible]   BS      .20000         .20000-       NONE       NONE        .
      9   [illegible]   BS      .15000         .15000-       NONE       NONE        .
     10   [illegible]   BS      .25000         .25000-       NONE       NONE        .
     11   [illegible]   BS      .15000         .15000-       NONE       NONE        .
     12   [illegible]   BS      .20000         .20000-       NONE       NONE        .
     13   [illegible]   BS    [illegible]    [illegible]     NONE       NONE        .
     14   [illegible]   BS      .15000         .15000-       NONE       NONE        .
     15   [illegible]   BS    [illegible]    [illegible]     NONE       NONE        .
     16   [illegible]   BS    [illegible]    [illegible]     NONE       NONE        .
     17   [illegible]   BS    [illegible]    [illegible]     NONE       NONE        .
     18   [illegible]   BS      .15000         .15000-       NONE       NONE        .
     19   [illegible]   BS        .              .           NONE       NONE        .
     20   [illegible]   BS      .23130         .23130-      .50000      NONE        .
     21   [illegible]   BS    [illegible]      .31130-      .50000      NONE        .
     22   [illegible]   BS      .76640         .26640-      .50000      NONE        .
     23   [illegible]   BS    [illegible]    [illegible]    .50000      NONE        .
     24   [illegible]   BS      .84435         .34435-      .50000      NONE        .
     25   [illegible]   BS    [illegible]    [illegible]    .50000      NONE        .
     26   [illegible]   BS      .73710         .23710-      .50000      NONE        .
     27   [illegible]   BS    [illegible]    [illegible]    .50000      NONE        .
     28   [illegible]   BS      .72360         .22360-      .50000      NONE        .
     29   [illegible]   BS    [illegible]    [illegible]    .50000      NONE        .
     30   [illegible]   BS      .82990         .32990-      .50000      NONE        .
     31   [illegible]   BS    [illegible]    [illegible]    .50000      NONE        .
     32   [illegible]   BS      .87020         .37020-      .50000      NONE        .
     33   [illegible]   BS    [illegible]    [illegible]    .50000      NONE        .
     34   [illegible]   BS      .91130         .41130-      .50000      NONE        .
     35   [illegible]   BS    [illegible]    [illegible]    .50000      NONE        .
     36   [illegible]   BS        .              .            .         NONE        .
     37   [illegible]   UL     4.00000           .           NONE      4.00000     .55000-
```

REPORT 7.3.1



```
          [illegible]
[illegible], TIME      0.12
   NAME      = [illegible]

[illegible]      [illegible]       [illegible]          [illegible]         PARAM =      .
TIME =   [illegible]  MINS.
  [illegible]                              [illegible] ....2[illegible]   ELEMENTS ....71   RECORDS .....1   ITERATION .....47
  BASIS ---- NO.OF ROWS ....37   LOGICALS .......26   STRUCTURALS .....1   ELEMENTS ....41
  [illegible] ........[illegible]   [illegible] .....[illegible]   ETA-VECTORS ....1[illegible]   ELEMENTS ....22   RECORDS .....1   TIME TAKEN   0.00

[illegible]      [illegible]       [illegible]          [illegible]         PARAM =      .
TIME =   [illegible]  MINS.
    [illegible]  [illegible]  [illegible]  [illegible]  [illegible]  [illegible]  [illegible]
    [illegible]  [illegible]   OUT     IN      COST      VALUE     VALUE
 *   [illegible]                                          [illegible]   .20000
 *   49      0      67      85   .03500-   13.2510    .40000
 *   [illegible]
 *   51      0     104      74   .00250-   13.1560   1.20000
 *   [illegible]
 *   53      0      [illegible]   114   .           13.[illegible]   1.[illegible]
 *   [illegible]
 *   55      0     1[illegible]   41   .[illegible]-   12.[illegible]   2.[illegible]
 *   [illegible]
 *   57      0      87     102   .[illegible]-   12.[illegible]   2.40000
 *   [illegible]
 *   59      0     127      58   .[illegible]-   12.[illegible]   3.00000
 *   [illegible]
 *   61      0     11[illegible]   62   .[illegible]-   12.[illegible]   3.40000
 *   [illegible]
 *   63      0      [illegible]   101   .11100-   12.[illegible]   4.20000
 *   [illegible]
 *   [illegible]   0     1[illegible]   47   .[illegible]-   12.1320   4.[illegible]
  [illegible]

  [illegible]                              [illegible] ....[illegible]   ELEMENTS ...14[illegible]   RECORDS .....1   ITERATION .....[illegible]
  BASIS ---- NO.OF ROWS ....37   LOGICALS .......26   STRUCTURALS .....1   ELEMENTS ....41
  [illegible] ........[illegible]   [illegible] .....[illegible]   [illegible] ....14   ELEMENTS ....22   RECORDS .....1   TIME TAKEN   0.00

[illegible]      [illegible]       [illegible]          [illegible]         PARAM =   4.[illegible]
TIME =   [illegible]  MINS.
    [illegible]  [illegible]  [illegible]  [illegible]  [illegible]  [illegible]  [illegible]
    [illegible]  [illegible]   OUT     IN      COST      VALUE     VALUE
 *   [illegible]
 *   [illegible]   0      [illegible]   44   .[illegible]-   11.[illegible]   [illegible]
 *   [illegible]
 *   [illegible]   0      94      70   .[illegible]-   11.[illegible]   5.[illegible]
 *   [illegible]
```

REPORT 7.3.2a



```
          TEST R H S

    ITER   NUMBER  VECTOR  VECTOR  REDUCED   FUNCTION   PARAM
   NUMBER  NONOPT   OUT      IN     COST      VALUE     VALUE
 *   71      0      120     [illegible]  [illegible]  11.5[illegible]  6.00000
 *   72      0       61      43  .02230-   11.5220    6.20000
 *   73      0       67     121  [illegible]  11.42[illegible]  6.40000
 *   74      0      121      67  .00120-   11.3160    6.60000
 *   75      0       67      69  [illegible]  11.20[illegible]  6.80000
 *   76      0       23      53  .01767-   11.1028    6.57778
 *   77      0  [illegible]  56  [illegible]  [illegible]  7.17778
 *   78      0       46      73  .01510-   10.8488    7.37778
 *   79      0       73     117  [illegible]  10.7056  7.57778
 *   80      0      113      83  .01300-   10.5578    7.77778
 *   81      0  [illegible]  39  [illegible]  10.3908  7.97778
 *   82      0       39      96  .01600-   10.2348    8.17778
 *   83      0  [illegible]  120  [illegible]  [illegible]  8.37778
 *   84      0      123     13[illegible]  .01520-   9.46691   8.57778
 *   85      0      12[illegible]  [illegible]  [illegible]  9.72191   8.77778
   INVERT DEMANDED AFTER  20 MAJOR/  20 MINOR ITERATIONS - CLOCK CONTROL

   INVERT CALLED     TIME   0.17   CURRENT INVERSE ---- ETA-VECTORS ....38   ELEMENTS ...162   RECORDS .....1   ITERATION .....85
   BASIS ---- [illegible] ....[illegible]   LOGICALS .......[illegible]   STRUCTURALS ......[illegible]   ELEMENTS ....49
   INVERSE -- NUCLEUS ........0   TRANSFORMED .....0   ETA-VECTORS ....18   ELEMENTS ....26   RECORDS .....1   TIME TAKEN   0.00

   PARAPHS      OBJ = FULLOBJ      RHS = RHS          CHCOL = RH2          PARAM =    9.77778

   TIME =    0.18  MINS.

    ITER   NUMBER  VECTOR  VECTOR  REDUCED   FUNCTION   PARAM
   NUMBER  NONOPT   OUT      IN     COST      VALUE     VALUE
 *   86      0       30      16  .33300-   9.57625    9.93862
 *   87      0       24     118  .01700-   9.50691    9.01752
 *   88      0       33     127  .21533-   9.46128    9.07413
 *   89      0  [illegible]  1[illegible]  [illegible]  [illegible]  9.1[illegible]
 *   90      0       35     106  .04400-   9.15858    9.32462
 *   91      0  [illegible]  [illegible]  [illegible]  9.1[illegible]  9.[illegible]
 *   92      0       23      62  .00201-   9.13279    9.34467
 *   93      0       22  [illegible]  [illegible]  9.10[illegible]  9.[illegible]
 *   94      0       26     1[illegible]0  .02931-   6.91186    9.50963
 *   95      0       31  [illegible]  [illegible]  [illegible]  9.[illegible]
 *   96      0       27      45  .11403-   6.58373    9.74333
 *   97      0       21  [illegible]  [illegible]  [illegible]  9.70165
 *   98      0       23     112  .33130-   9.36626    9.95311
 *   99      0  [illegible]   72  .01100-   [illegible]  9.99327
   PARAMETER AT USERS MAXIMUM
```

REPORT 7.3.2b

```
              TEST R.H.S
 SOLUTION   (OPTIMAL)
 TIME =   0.10 MINS. ITERATION NUMBER =     99

         ...NAME...    ...ACTIVITY...      DEFINED AS.......................

         FUNCTIONAL           0.18577      FULLOBJ
         RESTRAINTS                        RHS        +     10.00000  RH2
         SUPPORT....                       A
```

REPORT 7.3.2c

```
        TEST R H 3

SECTION 1 - ROWS

NUMBER ...ROW.. AT ...ACTIVITY... SLACK ACTIVITY ..LOWER LIMIT. ..UPPER LIMIT. .DUAL ACTIVITY

   1  [illegible]  BS    .15677        .15677-        NONE          NONE          .
   2  PHI001   BS    .09415        .09415-        NONE          NONE          .
   3  [illegible]  BS    .15677        .15677-        NONE          NONE         1.00000
   4  PHI002   BS    .09792        .09792-        NONE          NONE          .
   5  [illegible]  BS    [illegible]   [illegible]    NONE          NONE          .
   6  PHI004   BS    .15555        .15555-        NONE          NONE          .
   7  [illegible]  BS    [illegible]   [illegible]    NONE          NONE          .
   8  PHI006   BS    .06757        .06757-        NONE          NONE          .
   9  [illegible]  BS    [illegible]   [illegible]    NONE          NONE          .
  10  PHI008   BS    .08843        .08843-        NONE          NONE          .
  11  [illegible]  BS    [illegible]   [illegible]    NONE          NONE          .
  12  PHI010   BS    .07712        .07712-        NONE          NONE          .
  13  [illegible]  BS    [illegible]   [illegible]    NONE          NONE          .
  14  PHI012   BS    .05290        .05290-        NONE          NONE          .
  15  [illegible]  BS    [illegible]   [illegible]    NONE          NONE          .
  16  PHI014   BS    .13583        .13583-        NONE          NONE          .
  17  [illegible]  BS    [illegible]   [illegible]    NONE          NONE          .
  18  PHI016   BS    .05994        .05994-        NONE          NONE          .
  19  [illegible]  BS     .             .             NONE          NONE          .
  20  USER001  LL    .50000         .             .50000        NONE         .31538
  21  [illegible]  LL    .50000         .             .50000        NONE         .29222
  22  USER003  LL    .50000         .             .50000        NONE         .29278
  23  [illegible]  LL    .50000         .             .50000        NONE        [illegible]
  24  USER005  LL    .50000         .             .50000        NONE         .78166
  25  [illegible]  BS    [illegible]   [illegible]    .50000        NONE          .
  26  USER007  LL    .50000         .             .50000        NONE         .26627
  27  [illegible]  LL    .50000         .             .50000        NONE         .21947
  28  USER009  LL    .50000         .             .50000        NONE         .89803
  29  [illegible]  LL    .50000         .             .50000        NONE        [illegible]
  30  USER011  LL    .50000         .             .50000        NONE         .35195
  31  [illegible]  LL    .50000         .             .50000        NONE         .22981
  32  USER013  LL    .50000         .             .50000        NONE         .60146
  33  [illegible]  LL    .50000         .             .50000        NONE         .63667
  34  USER015  LL    .50000         .             .50000        NONE         .57994
  35  [illegible]  LL    .50000         .             .50000        NONE        [illegible]
  36  USER017  BS     .             .              .            NONE          .
  37  [illegible]  UL   1.50000         .             NONE         1.50000      [illegible]
```

REPORT 7.3.3

REFERENCES

Chapter Seven

[1] Churchman, C. West; Ackoff, Russell L.; and Arnoff, E. Leonard. Introduction to Operations Research. New York: John Wiley and Sons, 1957.

[2] Hadley, George. Non Linear and Dynamic Programming. Reading, Massachusetts: Addison-Wesley Publishing Co., 1964.

[3] IBM Bulletin No. H20-0476-0. Mathematical Programming System/360 (360A -CO- 14X) Linear and Separable Programming--User's Manual, 1967.

[4] IBM Bulletin No. H20-0136-3. Mathematical Programming System/360: Application Description, 1968.

[5] Stankard, Martin F.; Maier-Rothe, Christoph; and Gupta, Shiv K. "Choosing Between Multiple Objective Alternatives: A Linear Programming Approach." Technical Paper, Management Science Center, University of Pennsylvania, Philadelphia, December 1968.

BIBLIOGRAPHY

Ackoff, Russell L. *Scientific Method: Optimizing Applied Research Decisions.* New York: John Wiley and Sons, 1962.

______, and Sasieni, Maurice W. *Fundamentals of Operations Research*. New York: John Wiley and Sons, 1968.

Adams, E. N. "Reflections on the Design of a CAI Operating System." IBM Research. Yorktown Heights, New York: Report No. RC 1745, January 23, 1967.

______. "Roles of the Electronic Computer in University Instruction." IBM Research. Yorktown Heights, New York: Report No. RC 1530, October 29, 1965.

______. "Computer-Assisted Instruction." *Computers and Automation*, Volume 15, No. 3, March 1966.

______; Morrison, H. W.; and Reddy, J. M. "Conversation with a Computer as a Technique of Language Instruction." *The Modern Language Journal*, Volume LII, No. 1, January, 1968, pp. 3 - 11.

Atkinson, Richard. "Computerized Instruction and Learning Process." *American Psychologist*, Volume 23, No. 4, April 1968, pp. 225 - 239.

______, and Hansen, Duncan N. "Computer Assisted Instruction in Initial Reading: The Stanford Project." *Reading Research Quarterly*, 1966, Volume 2, pp. 5 - 25.

______, and Wilson, H. A. "Computer Assisted Instruction." *Science*, Volume 161, No. 3849, pp. 73 - 77, 1968.

Batchelder, William H. "A Mathematical Analysis of Multi-Level Verbal Learning." Technical Report No. 104, August 9, 1966, Institute for Mathematical Studies in the Social Sciences, Stanford University, Stanford, California.

Becker, James L. A Programmed Guide to Writing Auto Instructional Programs. Camden, New Jersey: R.C.A. Service Company, 1963.

Bitzer, D. L.; Hicks, B. L.; Johnson, R. L.; and Lyman, Elisabeth R. "The PLATO System: Current Research and Developments." IEEE Transactions on Human Factors in Electronics, Volume HFE-8, No. 2, June 1967, pp. 64 - 69.

Charp, Sylvia, and Hayman, John L., Jr. "The Research Design for the Philadelphia Program in Computer-Assisted Instruction." School District of Philadelphia Division of Instructional Systems, Mineographed, 1967.

Churchman, C. West; Ackoff, Russell L.; and Arnoff, E. Leonard. Introduction to Operations Research. New York: John Wiley and Sons, Inc., 1957.

Cogswell, John F., and Coulson, John E. "Effects of Individualized Instruction on Testing." S.D.C. Report No. SP-1829, Santa Monica, California, February, 1966.

Coulson, John E., ed. Programmed Learning and Computer Based Instruction. New York: John Wiley and Sons, Inc., 1962.

_____; Bushnell, Don D.; and Cogswell, John E. "Computer Based Instruction Systems." In Applied Programmed Instruction. Margulies and Eigen (ed.). New York: John Wiley and Sons, Inc., 1962, pp. 106 - 119.

_____, et al. "Effects of Branching in a Computer-Controlled Auto-Instructional Device." Journal of Applied Psychology, No. 46, pp. 389 - 392, 1962.

Cox, D. R., and Smith, Walter L. Queues. London: Methuen's Monographs on Applied Probability and Statistics, Methuen & Co., Ltd., 1961.

Crowder, Norman A. "Simple Ways to Use the Student Response for Program Control." In Applied Programmed Instruction. Margulies/Eigen (ed.). New York: John Wiley & Sons, Inc., 1962.

_____. "Automatic Tutoring by Intrinsic Programming." In Teaching Machines and Programmed Instruction. Lumsdaine and Glaser, (ed.), 1960.

Dear, R. E.; Silberman, H. F.; Estavan, D. P.; and Atkinson, R. C. "An Optimal Strategy for the Presentation of Paired Associates Items." Behavioral Science, Volume 12, No. 1, January 1967, pp. 1 - 13.

Engel, Gerald L. "Computer Assisted Instruction: A Selected Bibliography and KWIC Index." Dahlgren, Virginia: U. S. Naval Weapons Laboratory, Report No. K-3/67, January, 1967.

Feingold, Samuel L. "PLANIT--A Flexible Language Designed for Computer--Human Interaction." Fall Joint Computer Conference Proceedings. American Federation of Information Processing Societies, pp. 545 - 551.

_____. "PLANIT--A Language for CAI." Datamation, September, 1968, pp. 41 - 47.

Feldman, Julian. "Computers, Instruction and Learning." 15 February 1967 Address to the Third Annual Meeting of the Information Processing Association of Israel, Weizmann Institute, Rehovot.

Feller, William. An Introduction to Probability Theory and Its Applications, Volume I. New York: John Wiley and Sons, Inc., 1965.

_____. An Introduction to Probability Theory and Its Applications, Volume II. New York: John Wiley and Sons, 1965.

Feurzeig, W. "New Instructional Potentials of Information Technology." IEEE Transactions on Human Factors in Electronics, Volume HFE-8, No. 2, June 1967.

Frye, Charles H. "CAI Languages: Capabilities and Applications." Datamation, September 1968, pp. 34 - 37.

Gage, N. L., (ed.). Handbook of Research on Teaching. Chicago: Rand McNally, 1963.

Gagné, R. M. "The Acquisition of Knowledge." Psychological Review, 1962, 62, 313 - 322.

_____. The Conditions of Learning. New York: Holt Rinehart and Winston, Inc., 1965.

Galanter, E. H., (ed.). Automatic Teaching: The State of the Art. New York: John Wiley & Sons, 1959.

Gentile, J. Ronald. "The First Generation of Computer Assisted Instruction Systems: An Evaluative Review." A.V. Communications Review, Volume 15, No. 1, 1967, pp. 23 - 53.

Geoffrion, Arthur M. "Stochastic Programming with Aspiration or Fractile Criteria." Management Science, Volume 13, No. 9, May 1967, pp. 672 - 679.

_____. "Solving Bicriterion Mathematical Programs." Journal of Operations Research Society of America, Volume 15, No. 1, January - February, 1967, pp. 39 - 54.

Glassman, Jerrold. Programmed Reading. New York: Globe Book Co., 1966.

Groen, G. J., and Atkinson, R. C. "Models for Optimizing the Learning Process." Technical Report No. 92, February 11, 1966, Institute for Mathematical Studies in The Social Sciences, Stanford University, Stanford, California.

Project GROW. "Printed Reports Produced by Project GROW." Unpublished memorandum, mineographed.

Grubb, Ralph E. "Resources Needed for CAI." Programmed Learning and Educational Technology, Volume 5, No. 1, January 1968, pp. 18 - 24.

_____. "Programmed Learning." Educational Technology, Volume 5, 38, 1968.

Hadley, George. Non Linear and Dynamic Programming. Reading, Massachusetts: Addison-Wesley Publishing Co., 1964.

Hall, Arthur. A Methodology for Systems Engineering. Princeton: Van Hostrand, 1962.

Hartman, T. F. "Computer Assisted Instruction." Audio Visual Instruction, January 1966, Volume 11, No. 1, pp. 22 - 23.

_____, and Schoonard, J. W. "A Computer Assisted Instruction Statistics Problem Laboratory 1: Design and Embodiment." IBM Research, Report RC 1855, June 30, 1967.

Hickey, Albert E. "Computer Assisted Instruction, A Survey of the Literature." 2nd Edition. Newbury Port, Massachusetts: Entelek, Inc., 1967.

_____. "Computer Assisted Instruction, A Survey of the Literature." 3rd Edition. Newbury Port, Massachusetts: Entelek, Inc., 1968.

Hilgard, Ernest R. Theories of Learning. New York: Appleton Century Crofts, Second Edition, 1956.

_____, and Atkinson, Richard C. Introduction to Psychology. 4th Edition. New York: Harcourt, Brace and World, Inc., 1967.

Hitch, Charles J., and McKean, Ronald N. The Economics of Defense in the Nuclear Age. New York: Atheneum, 1967.

Howard, Ronald A. "Dynamic Inference." Journal of Operations Research Society of America, Volume 13, No. 5, 1965.

Hughes, J. L. Programmed Instruction for Schools and Industry. Chicago: Science Research Associates (print.), 1962.

IBM 1401 or 1440. "Operating System Computer Assisted Instruction." Form C24-3253-0, Endicott, New York: 1964.

IBM Research Reports. "Computer Assisted Instruction." Yorktown Heights, New York: Volume 2, No. 1, January 1966.

IBM Computing Report. "A Laboratory for Learning." Volume IV, No. 4, September 1968, pp. 8 - 11.

Kagan, Jerome. "Reflection--Impulsivity and Reading Ability in Primary Grade Children." Child Development, Volume 36, September 1965.

Karush, William, and Dear, R. E. "Optimal Strategy for Item Presentation in a Learning Process." Management Science, Volume 13, No. 11, July 1967, pp. 773 - 785.

_____. "Optimal Stimulus Presentation Strategy for a Stimulus Sampling Model of Learning." Journal of Mathematical Psychology, Volume 3, 1966, pp. 19 - 47.

Kay, Henry; Dodd, Bernard; and Sime, Max. Teaching Machines and Programmed Instruction. Penguin Books, Baltimore, Maryland: 1968.

Kempthorne, Oscar. The Design and Analysis of Experiments. New York: John Wiley and Sons, Inc., 1952.

Kopstein, Felix F., and Seidel, Robert J. "Computer Administered Instruction Versus Traditionally Administered Instruction: Economics." A.V. Communications Review, Volume 16, No. 2, Summer 1968, pp. 147 - 175.

Lumsdaine, A. A., and Glaser, Robert, (Ed.). Teaching Machines and Programmed Learning. Washington, D.C.: Department of Audio-Visual Instruction, National Education Association, 1960.

Margulies, Stuart, and Eigen, Lewis D., (ed.). Applied Programmed Instruction. New York: John Wiley and Sons, 1962.

Maier-Rothe, Christoph. Elements of a Normative Theory of the Firm. Ph. D. Thesis in Operations Research, University of Pennsylvania, Philadelphia, 1969; in preparation.

Markle, Susan Mayer. "Empirical Testing of Programs." Programmed Instruction. The 66th Yearbook of the National Society for the Study of Education, Part 2, 1967, pp. 104 - 138.

Mathematical Programming System/360: Application Description. IBM Bulletin No. H20-0136-3, 1968.

Mathematical Programming System/360 (360A -CO- 14X) Linear and Separable Programming--User's Manual. IBM Bulletin No. H20-0476-0, 1967.

Mayer, Sylvia R. "Computer Based Subsystems for Training the Users of Computer Systems." IEEE Transactions on Human Factors in Electronics, Volume HFE-8 No. 2, June 1967, pp. 70 - 74.

Miller, James G. "Living Systems, Structures and Processes: Cross Level Hypotheses." Behavioral Science, Volume 10, No. 4, October 1965.

Mitzel, Harold L., and Brandon, George L., et al. Experimentation with Computer Assisted Instruction in Technical Education: Semi Annual Progress Report. Pennsylvania State University, Pennsylvania, Computer Assisted Instruction Laboratory, Report No. R-6, June 1967.

_____. Experimentation with Computer Assisted Instruction in Technical Education: Semi Annual Progress Report. The Pennsylvania State University, Pennsylvania, Computer Assisted Instruction Laboratory, June 30, 1968.

Morrison, Donald F. Multivariate Statistical Methods. New York: McGraw-Hill, 1967.

Morrison, H. W., and Adams, E. N. "Pilot Study of a CAI Laboratory in German." Modern Language Journal, Volume LII, No. 5, May 1969, pp. 280 - 287.

Niedermeyer, Fred C. "The Relevance of Frame Sequence in Programmed Instruction: An Addition to the Dialogue." A.V. Communications Review, Volume 16, No. 3, Fall 1968, pp. 301 - 317.

Pask, Gordon. "The Control of Learning in Small Subsystems of a Programmed Educational System." IEEE Transactions on Human Factors in Electronics, Volume HFE-8, No. 2, June 1968, pp. 88 - 93.

Quinn, E. M. "A CAI Reading Program--Preliminary Field Test." IBM Research, Yorktown Heights, New York: Research Note No. NC 576, January 21, 1966.

_____, and Odeh, Anna. "Computer Assisted Instruction Arithmetic Drill and Practice Exercises Description of Organization and Logic." IBM Research, No. RC 1898, September 1968.

Ramo, Simon. "A New Technique of Education." In Teaching Machines and Programmed Instruction. Lumsdaine and Glaser (ed.), 1960.

Randall, R., and Blaschke, C. "Economics, Management and Public Policy." Educational Technology, Volume 8, No. 12, June 30, 1968.

Rath, Gustave J. "The Development of Computer Assisted Instruction." IEEE Transactions on Human Factors in Electronics, June 1967.

_____. "Non CAI Instruction Using Computers and Non Instructional Uses of CAI Computers." Educational Technology, Volume 8, No. 3, pp. 11 - 13.

_____. "Computer Teaching." IEEE Transactions on Human Factors in Electronics, June 1967, Volume HFE-8, No. 2, p. 59.

Rogers, James L. "Current Problems in CAI." Datamation, September 1968, pp. 28 - 33.

Scheflen, Albert E. "Human Communication: Behavioral Programs and their Integration in Interaction." Behavioral Science, Volume 13, No. 1, January 1968.

Schurdals, J. J. "An Approach to the Use of Computers in the Instructional Process and an Evaluation." IBM Research Yorktown Heights, New York: Report No. RC 1432, July 6, 1965.

Schwartz, H. A., and Long, H. S. "Instruction by Computer." Datamation, September 1966, pp. 73 - 79.

Seidel, Robert J., and Kopstein, Felix F. "A General Systems Approach to the Development and Maintenance of Optimal Learning Conditions." Alexandria, Virginia: George Washington University, Human Resources Research Office, September 1967.

Silvern, Gloria M., and Silvern, Leonard C. "Computer Assisted Instruction: Specification of Attributes for CAI Programs and Programmers." 1966 Proceedings of 21st National ACM Conference, pp. 57-62.

______, and Silvern, Leonard C. "A Graduate Level University Course in Methods of Computer Assisted Instruction." IEEE Transactions on Human Factors in Electronics, Volume HFE-8 No. 2, June 1967, pp. 94 - 102.

Sisson, Roger L. "Sequencing Theory." Chapter 7 in Russell L. Ackoff (ed.), Progress in Operations Research, Volume I, New York: John Wiley and Sons, 1961.

Skinner, B. F., and Holland, James G. "The Use of Teaching Machines in College Instruction." In Teaching Machines and Programmed Instruction. Lumsdaine and Glaser (ed.), 1960.

_____. "Teaching Machines." In Teaching Machines and Programmed Learning. Lumsdaine and Glaser (ed.), 1960

Skinner, B. F. *The Technology of Teaching*. New York: Appleton Century Crofts, Division of Meredith Corporation, 1968.

_____. "The Science of Learning and the Art of Teaching." In *Teaching Machines and Programmed Learning*. Lumsdaine and Glaser (ed.), 1960.

Smallwood, Richard D.; Weinstein, Iram; and Eckles, James E. "Final Report on Office of Naval Research Contract Nonr-222(84)." March 15, 1967, Institute in Engineering-Economic Systems, Stanford University, Stanford, California.

_____. *A Decision Structure for Teaching Machines*. Cambridge, Massachusetts: The M. I. T. Press, 1962.

Stankard, Martin F.; Maier Rothe, Christoph; and Gupta, Shiv K. "Choosing Between Multiple Objective Alternatives: A Linear Programming Approach." Management Science Center, University of Pennsylvania, Philadelphia, Technical Paper, December 1968.

Stolurow, L. M. "What is Computer Assisted Instruction." *Education Technology*, Volume 8, No. 15, August 15, 1968, pp. 10 - 11.

_____. *Teaching by Machine*. Washington, D. C.: U. S. O. E. Monograph No. 6, U. S. Government Printing Office, 1961.

_____. "Systems Approach to Instruction." 1965, University of Illinois Training Research Laboratory Technical Report No. 7.

_____. "Computer-Based Instruction: Psychological Aspects and Systems Conception of Instruction." Report on ONR Contract, N00014-67-A-0298-0003, 1967.

Suppes, Patrick, and Atkinson, Richard. *Markov Learning Models for Multi person Interactions*. Stanford University Press, 1960.

Zinn, Karl L. "Instructional Use of Computer Systems." Datamation. September 1968, pp. 23 - 27.

_____. "Guide to Literature in the Area of Instructional Uses of Computers." Educational Media and Technology Clearinghouse, Stanford, California, 94305.

GLOSSARY

algorithm - a computational procedure for solving a problem. When properly applied, an algorithm always produces a solution to the problem.

authors - those who design instructional material for presentation by the computer.

basic feasible solution - for any linear programming problem only a small subset of all feasible solutions are candidates for the optimal solution. These candidates are called basic feasible solutions.

branching - altering the course of a set of instructions by switching when some predesignated event occurs.

CAI programming language - a user oriented language for programming a computer to display curriculum, receive and process responses, and for branching to appropriate portions of the curriculum as required by the logic of instruction.

capital intensive - an economic term used to describe a process in which capital (machines or previously constructed facilities) is the predominant factor of production.

cognitive - mental processes that refer to knowing and thinking.

console - the desk at which a student interacts with the computer. A console is equipped with a television screen, a keyboard and a light pen.

contingencies of reinforcement - the conditions of learning propounded by B. F. Skinner. The three contingencies of reinforcement are (1) the conditions surrounding an organism's behavior (2) the behavior itself and (3) the consequences of the behavior.

debug - to search for and correct errors in a computer program.

dynamic programming - a class of mathematical optimization procedures based on Richard Bellman's PRINCIPAL OF OPTIMALITY. Dynamic Programming is conceptually powerful but suffers, in most practical applications, from its requirements for enormous computer storage facilities.

feasible solution - any set of values for variables that satisfy the constraints of a mathematical programming problem. A feasible solution is not necessarily an optimal solution to the problem.

feedback - in programmed instruction--providing the student with information on the correctness of his last output or response. The feedback may be designed to correct a student's incorrect response.

frame - the smallest unit of programmed instruction. A frame consists of an instructional stimulus, an opportunity for response, and a feedback portion.

heuristic - a guide to finding a solution to a problem that cannot be proved to always result in a solution.

Kuhn-Tucker conditions - state the mathematical requirements for finding the maximum or minimum value of a linear or nonlinear function subject to a set of constraints.

labor intensive - an economic term that describes a process in which labor is the primary factor of production.

latency - the time from the display of an instructional stimulus to the start of the student's response. In latency data gathered by the computer system the time taken to make the response is also included in the latency values.

learning - a semi-permanent change in the behavior of an organism.

light pen - a stylus that can sense the coordinates of a point on the screen of a console. It uses the light from the cathode ray tube (T.V. tube) to determine where on the face of the tube the stylus is being held.

linear programming - mathematical techniques for optimizing a linear function of several variables subject to linear inequality constraints on some or all of the variables. Recently this term has been used as a name for the simplest form of programmed instruction.

model - an idealized representation that demonstrates the relationships between relevant variables. Models are used to better understand and control a real situation.

motivation - the value that an individual places on the result of some behavior. The value is generally described in terms of need satisfaction.

Northwest Corner procedure - a simple algorithm for solving allocation problems subject to constraints. Allocations of resources to activities are made by working from the upper left (Northwest) corner of a tableau.

objective function - in mathematical optimization--the function to be optimized (maximized or minimized).

off-line - processes performed outside of the operation of the central processor of a computing system.

on-line - processes performed under the control of the central processor of a computing system. On-line operations proceed without external intervention in the activities of the computing system.

operant - behavior emitted by an organism not necessarily in response to a stimulus. Operants are chance behavior.

operant conditioning - a teaching process whereby selected operants are made more probable by carefully rewarding an organism upon the completion of the desired operant.

operational definition - an explicit statement of the conditions and operations by which a concept may (ideally) be identified.

operations research - the use of a scientific approach to solving management control problems within large or complex organizations.

parametric studies - systematic variation of the value of a coefficient in a mathematical model in order to study the dependence between the solution of the model and the coefficient.

programmable counters - (also author counters) software devices that may be used by authors to count the number of times that some event occurs. For example a counter may be established to keep track of the number of times that a student responds incorrectly.

programmed instruction - teaching that uses a step by step method of presenting basic elements of subject matter to a student. Programmed Instruction usually alters the course of instruction depending upon the results of past instruction (branching).

programmed text book - a text book that is designed explicitly to teach. It presents information, requires a response of the reader and then provides him with feedback.

reinforcement - any consequence of an organism's behavior that increases the likelihood that the organism will repeat that behavior.

remediation - instructional action taken because a learner has made an inappropriate response in a frame. Remediation decreases the likelihood of the inappropriate response.

response - a behavior or an activity that may be the result of some physical or mental stimulus.

sensitivity analysis - the investigation of relationships between the solution of a problem and errors in formulation of the problem.

separable programming - a technique for optimizing certain non linear programming problems. A new linear programming problem is derived from the original problem. Each non linear term in one variable is replaced by several approximating variables.

shadow price - the increase in value of the objective function that will result from relaxing a constraint on an optimal solution.

simplex algorithm - a mathematical procedure for solving a broad class of linear programming problems.

software - all types of programs required for directing the operation of a computing system. Software is written in a formal language which can be processed by the computer.

stimulus - an observable event that is the occasion for an organism's behavior.

system - an interacting group of elements. A computer system includes men and machines.

system counters - software devices which count the number of times that some event occurs during the operation of the system.

teaching - (ideally) a sufficient condition for learning. The arrangement of a student's environment so that his behavior is modified.

teaching point - the lowest level of detail in an outline of material to be presented by programmed instruction. Each frame of programmed instruction corresponds to a teaching point.

time sharing - the use of a single processing unit to serve many users simultaneously. Each user's demands are served by the system for only a fraction of real time.

trade-off - the amount of one valuable item that would be given up to acquire an amount of an alternative valuable item.

ABSTRACT

For a particular CAI system and course, this research answers the question, "How much computerized instruction should be given to each student in a group in order to achieve educational objectives for the group"? The work also illustrates the usefulness of operations research in solving complex educational management problems. The performance measure is the forecasted average achievement of the class on a final test over the CAI course material. This performance function is maximized subject to constraints on the allowable probability that each student might fail the final test and a constraint on total CAI console capacity.

The main elements of the research are: (1) a learning model which relates a student's time on the CAI course to expected final course achievement, (2) a procedure for forecasting the parameters of each student's learning model, and (3) a mathematical formulation so that standard methods of solution may be used to find a schedule that satisfies all of the constraints and that maximizes expected class average test score.

The models are tested on data from actual CAI operations in the Philadelphia School District's Project GROW and the Mathematical Programming System/360 is used to solve a case example.